ArGest® Backup

Producer's Edition 4.1.4 (Build 362)
Serial Number: 2000-0001
Copyright © 2004-2021, OWC, Inc.

ArGest® Backup
User's Guide

Version 4
Copyright © 2022, Other World Computing, Inc.

ArGest® Backup User's Guide

Other World Computing, Inc.., et al
Copyright © 2008-2022, Other World Computing, Inc., All rights reserved

Update Information
Other World Computing, Inc. will always work to insure that the data contained in this manual is kept up to date. You can always find the latest version at our website at

`http://www.argestsoftware.com/documentation.html`

Table Of Contents

SCREENSHOT NOTE -

Throughput this manual, screenshot images from all three platforms are used. Separate platform shots will be provided and indicated only when there is a specific platform difference that needs to be taken under consideration for the discussed operation(s).

1
What is ArGest Backup?

ArGest Backup - Wizard View
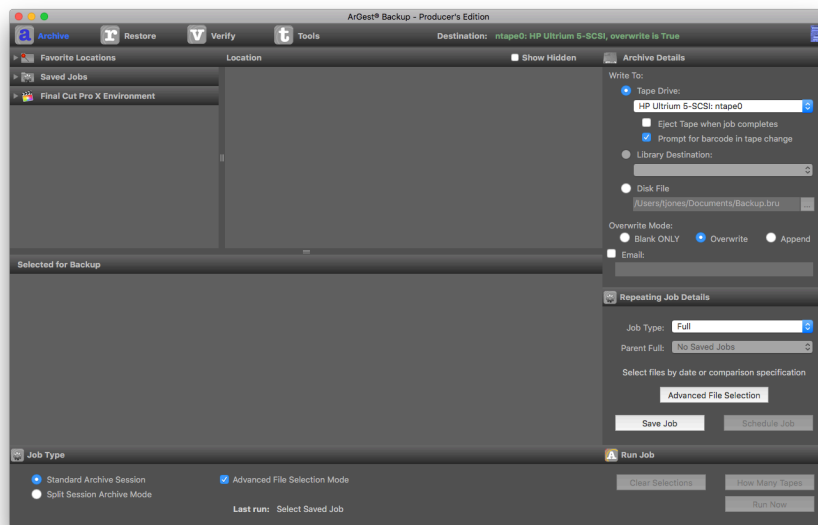
ArGest Backup - Advanced View

General Notes

ArGest Backup is a desktop-oriented, hands-on backup, archival, and recovery application designed designed to allow the most non-technical user to perform fully verified archival of personal, business, or production assets through an intuitive and easy to use drag, drop, and click operation.

ArGest Backup provides the highest level of data integrity by the use of TOLIS Group's BRU™ technology - an archival technology used for over 34 years in the Unix systems world providing backup and archival of data ranging from small office server environments to government forensic lab support and protection of critical project data for agencies such as NASA's Jet Propulsion Laboratories and Lawrence-Livermore National Labs.

ArGest Backup wraps the reliability and performance of the BRU engine within an easy to use interface that was designed to provide the highest levels of both performance and reliability in one easily learned archival application.

Device Compatibility

ArGest Backup provides support for all major modern tape technologies - LTO, DAT, and most other tape technologies - in both standalone and library arrangements while supporting all current I/O transport layers, including USB, Firewire, SCSI, SAS, SATA, Fibre Channel, and Thunderbolt. In each configuration, ArGest Backup will use the features of both the transport layer and the tape device technology to ensure high performance, ultra reliability, and low system resource utilization. ArGest Backup also supports disk targets, allowing you to use low cost removable disk devices for archive storage.

Depending on your license, ArGest Backup supports 1 or 2 tape devices and tape libraries and autoloaders with up to 24 tape slots. For disk storage, ArGest Backup will support the creation of archives on any mountable volume - including RDX, DVD-RAM, Firewire/USB/eSATA disk, SAN and NAS storage arrays and private or public cloud storage pools.

Performance

ArGest Backup has no upper data content size limit and can support single stream archives of 8TB or more. Conceptually, because of its 64 bit data handling abilities (even on 32 bit systems), ArGest Backup can process a single backup stream of up to 16 exabytes.

With high speed tape or disk technologies, ArGest Backup can easily write archives at over 500MB/sec. Of course, that number will be dependent on both the type and size of data being archived and the speed of your source storage devices, but consistent throughput rates of 310MB/Sec when writing to a SAS LTO-7/8 tape drives have been witnessed by many customers.

Device	Interface	Performance	Native Capacity
LTO-1	SCSI, FC	15MB/Sec, 54GB/Hr	100GB
LTO-2	SCSI, FC	30MB/Sec, 108GB/Hr	200GB
LTO-3	SCSI, SAS, FC	40-80MB/Sec, 144-288GB/Hr	400GB
LTO-4	SCSI, SAS, FC	40-110MB/Sec, 144-396GB/Hr	800GB
LTO-5	SAS, F-C	40-140MB/Sec, 144-504GB/Hr	1.5TB
LTO-6	SAS, F-C	40-160MB/Sec, 144-576GB/Hr	2.5TB
LTO-7	SAS, F-C	40-300MB/sec, 144-1,080GB/Hr	6TB
LTO-8 (M8)	SAS, F-C	40-300MB/sec, 144-1,080GB/Hr	9TB
LTO-8	SAS, F-C	40-300MB/sec, 144-1,080GB/Hr	12TB

Device Expected Average Performance Table

In the table above, the performance range on faster drives depends on your host and filesystem performance. The lower number is average for an older USB-2, SATA-1, or PATA-based system. The higher number may be even faster than indicated if you have a very high throughput SSD/NVMe or RAID-based filesystem utilizing FC or SAS for the transport layer.

Platform and Version Compatibility

Unlike many other backup and archival applications, ArGest Backup archives are compatible with all other BRU products on all platforms. Therefore, an archive created with ArGest Backup on Apple's macOS today will be recoverable with a future version of ArGest Backup or any other BRU product on any compatible platform. You can create archives of data using ArGest Backup for OS X and then restore that data with BRU Workstation on Oracle/Sun Solaris or even BRU Server for Microsoft Windows.

ArGest® Backup 4.0 is natively supported on Windows 7, 8.1, and 10, macOS 10.11 through 10.15 (Catalina), and 64bit Linux platforms using the GTK3 desktop graphics layer on X11 or Wayland servers.

A Word About Capacity and Compression

When considering media requirements for your tape-based backup operations, you must always use the ***Native Capacity*** rating for all tape technologies. Most tape device and media manufacturers market their technologies based upon a hypothetical 2:1 or even 2.5:1 compression value - meaning that for every two bytes of data in a file on your disk, only one byte of space is required on the tape. Unfortunately, these values are almost impossible to achieve unless you are working with a specially defined set of data files where the contents contain enough repetitive data elements to allow the compression algorithms to combine them to conserve space on the storage media. For even the best of day to day environments, that number is impossible to achieve.

In a normal office environment where your file data consists mainly of email, word processing documents, spreadsheets, and other text-based files, you may see your results reach 1.4:1 (meaning that the native, uncompressed 2.5TB capacity of an LTO-6 tape will appear to hold 3.5TB). However, if you are working with film or music content, your data has already been compressed by your file type definitions (BWAV, DXF, MOV, etc.) and you will only see 1:1 for the device's capacity (LTO-5 = 1.5TB/tape, LTO-6 = 2.5TB/tape, LTO-7 = 6TB/tape and LTO-8 = 12TB/tape).

This holds true for both BRU's software-based compression when writing to disk and all tape drive's hardware-based compression when writing to tape.

Fast Recovery

ArGest Backup uses a tape drive's built in physical block processing to keep track of where every file is located when written to a tape. This allows ArGest Backup to restore files written onto large capacity tapes - such as LTO-7 or LTO-8 media - in minutes. Even on archives consisting of multiple tapes, ArGest Backup will only ask for the tape or tapes that contain the data being restored and use the drive's fast seeking capabilities to locate a file directly rather than scanning through all tapes used in a given archive. That means that even on a fully loaded LTO-8 tape, restore of physical data will begin in 5 minutes if the tape must seek to the very last block; much more quickly if the starting point is much earlier on the tape.

Recover to an Alternate Location

Since you may need to recover data to an alternate system or location on the original system, ArGest Backup provides two methods for restoring to an alternate location:

Recovery trimming original path
This mode will trim any path elements from the selected data and restore the chosen files or folders directly into the alternate path. For example, a folder that was originally located at:

        `/Volumes/Storage 3/Project/Dailies`

Being recovered into

        `/Users/me/Project/`

Would result in a restored file layout of

        `/Users/me/Project/Dailies`

Recovery using full original path
This mode will retain the full path of the original files or folders and add them to the path selected as the recovery location. Using the previous example, the resulting recovered data would be located at:

        `/Users/me/Project/Volumes/Storage 3/Project/Dailies`

It's important to realize that in this second case, BRU will restore all the original properties and permissions and the resulting recovered files or folders will be hidden in the Finder since the original `/Volumes` folder is hidden in Finder by default.

Examples of files that exist under the `/Volumes` directory are external drives such as Xsan , FireWire , USB, SMB, AFS, NFS, SAS, eSATA, Fibre Channel RAID devices - basically any volume available to your system that is not the root or boot volume.

To locate the files after such a recovery has completed, you may either use the Finder's "Go to" command or the Terminal with the '`ls`' (ell ess) command.

To use Finder to get to the newly restored files, press ⌘-SHIFT-G (command-shift-g). When the "Go to..." window appears, type in the full path to the restore point and append `/Volumes` to the path. For example, if you restored your data to `/Users/me/Desktop`, then the full path you need to enter is:

        `/Users/me/Project/Dailies/Volumes/`

Then press [ENTER] and you will see the contents of the previously hidden folder. You may now move the restored data to a visible folder.

If you get an error, check your path and make sure capital letters are used when needed as well as making sure that the full path to the files was entered correctly.

The BRU Tape Format

BRU is not simply a variation of tar, or any other archival format, it's a completely different archive format definition. To ensure the highest level of recoverability, BRU archives contain far more metadata, or filesystem information about the data that was backed up. Additionally, rather than simply reading the data from your filesystem and copying it to the archive destination, BRU reads your data in 2 Kilobyte segments and performs additional checksum generation at that 2KB level.

While this operation results in a slight increase in the overall archive size when compared to the original data, the additional filesystem metadata and checksum information provided by this increase provides very robust verification and recovery capabilities that other backup applications lack.

Verify Capabilities

AnyTime Verify
Thanks to BRU's 2KB checksum mechanism, ArGest Backup is able to verify the contents of your archives at any time, not only immediately following the archive operation. This means that you can determine the validity of an archive at any time without requiring access to the original filesystem data. This also means that you can perform verify passes on archival media on different machines, or even on a different OS platform.

Comparison Verify
If you do have the need to compare your archive contents to the original filesystem data, you can also do this using ArGest Backup's Comparison Verify mode. In this mode, ArGest Backup will reread the archive contents and compare not only the file data, but also any permissions, ACL settings and Extended Attributes to those of the original filesystem. This mode is good for comparing a known good filesystem state to a filesystem suspected of corruption or possible tampering.

Special Files

BRU will save and restore all types of filesystems and files with their proper ownership, access attributes, creation dates, and modification dates. BRU can be used to move, or clone, an entire directory hierarchy from one system to another, with all files, including directories, block special files, character special files, fifos, hard links, and symbolic links reproduced with all attributes intact.

File Comparisons

BRU can compare the contents of an existing archive with the current online files, reporting all those online files that have been modified, have had their attributes changed, or have been removed since the archive was created. For example, using a reference archive of a standard UNIX distribution, BRU can be used to detect which files have been changed or are missing. This is a valuable capability when you encounter system problems and you suspect that the cause is corrupted or missing system files.

File Restore Overwrite Protection

By default, when extracting (restoring) files, BRU will not overwrite existing files on the destination volume with older files of the same name. This feature can be overridden when necessary.

Catalogs - What Are They?

To make locating data in an archive easier, ArGest Backup uses a combination of disk and tape-based catalogs which contain the full listing of every file and folder that was included in the archive. To insure availability, the catalogs are written both to your ArGest Backup system disk and following the archive on the tape. This allows you to easily transfer tapes to another system using the export catalog functionality or recover the catalogs in the event of a system failure using the ArGest Backup Import Tool.

Please note - the archive catalogs are **NOT** your archives, nor are they required to restore data from a BRU archive; they simply make it easier to search for a specific file or folder for performing a restore operation. While a catalog entry will tell you what's **IN** an archive, you still need the physical archive / tape to restore data.

For more detailed information of how the catalogs are managed in reference to your ArGest Backup environment, please refer to Appendix B - Tape / Archive / Catalog Design Notes later in this guide.

Archival Versus Backup

One of the poorly understood concepts when using a backup solution is the difference between Backup operations and Archival operations. The primary descriptive difference is that an Archival operation is defined as a job that copies data from one storage location to another with the intention of removing the original data from the original storage location. On the other hand, a backup is simply a copy of data written to a secondary storage location (i.e.: tape) that can be used to recover the original data in the event that the original data is lost or modified.

Additionally, while a single offline copy of files for day to day backup purposes may be sufficient, a single copy on tape should not be considered sufficient when creating a copy for archival. Since the purpose of an archival copy is to allow you to remove the original data from its online storage location while being able to recover the data in the future, having a single copy on tape (or any other storage device) is inviting disaster when the deleted data must be recovered.

The most important consideration if you are running Archival operations is that your tapes should be protected from overwrite as soon as possible after you've completed the job. There are two ways to do this:

1. Immediately flip the tape's write protect tab as each tape is filled if using a standalone drive or remove the media from the library once the job completes and write protect each tape. - use this mechanical feature to ArGest Backups "Overwrite tape only if blank" feature described below.

2. Use WORM media - once a WORM tape is written, it cannot be overwritten or erased and no user intervention is required to protect the data written to the tape.

In either scenario, the tapes used when running the Archival job are now protected from accidental overwrite or erasure. The only limit is that the non-WORM media can be reset to writeable if someone flips the write protect tab to the write-enabled position.

Tape Data Protection Options

ArGest Backup provides one additional feature to help deter overwriting previously written tapes in the overwrite preference setting "Overwrite tape only if blank"
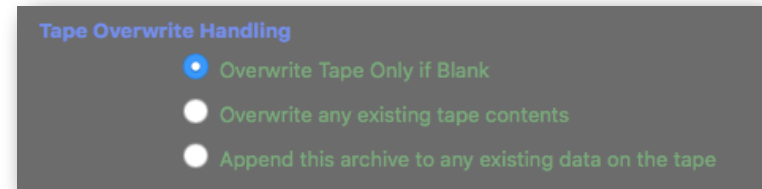
Tape Overwrite Preference Settings

This setting will not provide an option to continue writing to a tape that is not completely blank when inserted or loaded for the backup or archival operation.

Finally, be sure to label and store your archive job media properly. Do not mix it with your normal backup tapes and definitely keep it away from your blank media.

Definitions and Concepts

The following are terms that will be used through this manual and the associated command line argument when using BRU from a terminal.

backup | archive (-c option)

Used synonymously with backup. When you back up files you create an archive containing copies of the files. Archive is used both as a verb and a noun.

extract (-x option)

To restore from an archive.

full backup

A backup of all files on the system. Typically, this includes system files, system programs, and temporary or work files.

incremental backup (-n option)

An incremental backup copies (backs up) all files that have been modified (changed) since a given date. A combination of full system backups, usually at weekly intervals, and daily incremental backups is a common backup scheme. Incremental backups are smaller and therefore less time consuming than full backups to run and monitor. They also use less space on whatever media you use.

differential backup (-n option)

A differential backup includes all files that have been modified or created since the last full backup was performed. Differential backups differ from incremental backups in that files will be backed up multiple times even if they have not been modified since the previous differential backup.

partial backup (-QX and -X options)

A backup of only parts of the system. Typically a partial backup will not back up system files, the files that contain system programs, work files, etc. The files that are to be backed up may be specified by typing file names on the command line, by reading in a file that lists the files you want to back up, or by telling BRU to look at the `bruxpat` file. For a more complete description, refer to Chapter 15.

restore (-x option)

Extract files from an archive and return them to your normal filesystem.

inspect | verify (-i option)

BRU's primary verification method. This recalculates the checksums for the data blocks written to an archive to compare them against the checksums created with the backup was initially executed. Any differences reported could indicate a problem with the data included in the file listed.

differences | comparison (–d option)

This verification version will compare the files in an archive to their original version on disk. This mode is helpful for comparing a known-good backup against a system disk to check for tampering. The -d operation will report any differences that exist between the tape and the disk files.

table of contents (-t option)

To list the contents of an archive in a file-by-file listing.

Important Backup Data Selection Concepts
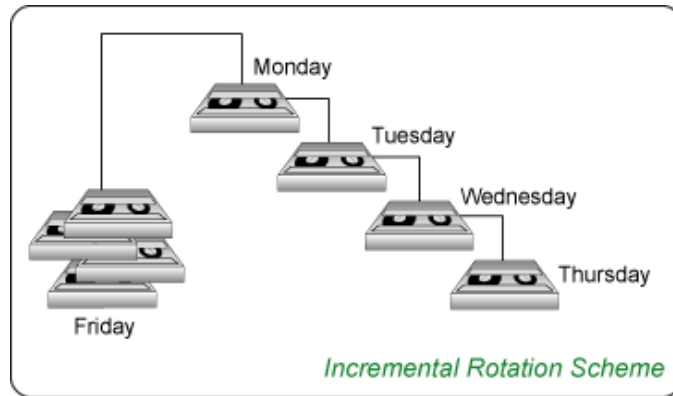
Incremental Rotation Scheme

An incremental backup is defined as a backup that includes files with a modification or creation date newer than the date-stamp of the last backup of its type, rather than the last full backup of its type. This means that the contents of a given incremental backup will probably remain relatively consistent from backup to backup.
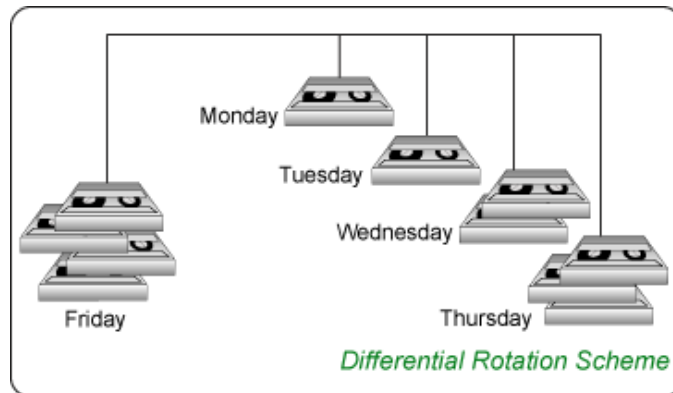
Differential Rotation Scheme

A differential backup is a backup that includes any files that have either a modification or creation date that is newer than the last FULL backup. We stress the word full because the date-stamp of the last FULL backup marker will be used for each differential backup. This means that each differential backup will be slightly larger than the last, so keep this in mind when defining the tape media used for the differential backups.

When deciding on an incremental or differential backup process, examine the amount of work that will be required to recover a given dataset. When using an incremental backup scheme, the restore process will require access to the most recent full backup and all of the incremental backups that have occurred since that full backup. For a differential schedule, the restore will require access to the last full backup and the most recent differential backup.

With that in mind, if you are using a tape library or disk stage for backup, the incremental process will be a very easy to implement strategy since archive maintenance and tracking is relatively automated. However, if you are using a standalone tape device, while the differential strategy may utilize more media, it will simplify the restore process.

<div align="right">

_2_
Installation, Licensing, and QuickStart

</div>

Installation

ArGest Backup is provided in a standard macOS DMG and Installer package, a Windows Installer EXE package, or as a Linux TGZ package format.

macOS 10.11 – 13

To install ArGest Backup, open the DMG and Drag the Application to the Application Folder alias. When you first launch the application, ArGest Backup will prompt for permission and install the system-critical components into `/usr/local/bin`. The remainder of the components are installed into

`/Library/Application Support/ArGest Backup`

and the UI components - ArGest Backup and bruTALK are installed into

`/Applications/ArGest Backup/.`

Microsoft Windows 7, 8.1, 10, and 11

Installation on Windows is a standard Windows EXE installation package. Once installed, the ArGest Backup application will be installed in

`C:\Program Files\ArGest Backup\`

and the operating environment will be created in

`C:\ProgramData\OWC\ArGest Backup\`

64bit Linux

Because there are so many Linux distributions, the ArGest Backup package is a gzipped tarball created using the absolute path flag. To install the solution - including the Desktop file for your

Applications menu, use the following command (assuming the TGZ file was downloaded to your Downloads folder):

`sudo tar -zxPf ~/Downloads/ArGest_Backup_4.*.tgz`

Note that the "P" (upper case P) argument is required for proper installation. Once completed, the ArGest Backup GUI application will be installed in

`/opt/ArGest Backup/`

the environment will be created in

`/usr/local/ArGest Backup/`

and the Open Desktop entry file will be added to

`/usr/local/share/applications/`

Your Applications menu will be updated automatically and the ArGest Backup application will be listed in the "Administration" section:
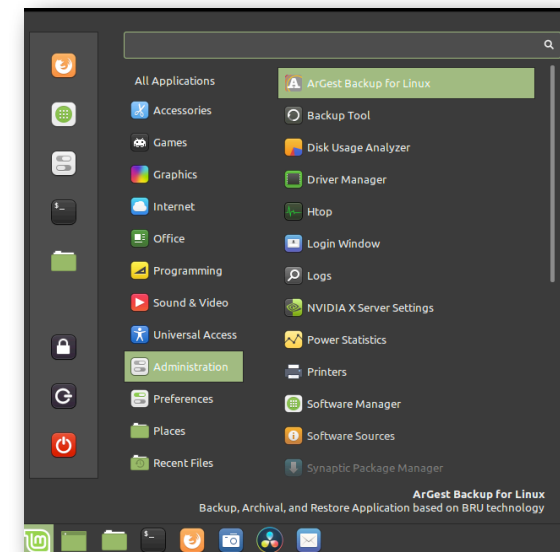
Linux Application Menu Entry

For each platform, the bru, brutalk, and other binary tools will be installed in a system PATH location for terminal or CLI access. On Windows, this is a Registry update that will require that you logout and back in as your user for the PATH change to take affect.

License Types

ArGest Backup is available in three license levels, or Editions - Desktop Edition, Workstation Edition, and Producer's Edition. The license that you purchase will determine which edition and any extensions that are appropriate for your specific installation.

Desktop Edition:

The Desktop Edition license provides all normal backup, archival, and restore functions, but is limited to creating disk-based backups, and writing to a single tape drive or a single RDX drive. It does not support any form of automation nor the Doubler Mode of operation. Specific notes or features relating to Desktop Edition will be indicated with the magenta circle-b logo.

Workstation Edition:

The Workstation Edition License provides all normal backup, archival, and restore functions, adding support for 2 tape drives, a single tape library of up to 2 drives and 25 slots (50 slot expansion is available), and 4 RDX drives. Specific notes or features relating to Workstation Edition will be indicated with the blue circle-b logo.

Producer's Edition:

The Producer's Edition License provides all normal backup, archival, and restore functions, adding support for 2 tape drives, a single tape library of up to 2 drives and 25 slots (50 slot expansion is available), and 4 RDX drives. Support for session or project file support and expansion licenses to support drag and drop of CMX3600 formatted EDL files are available. Specific notes or features relating to Producer's Edition will be indicated with the green circle-b logo.

ArGest Backup License Variants at a Glance

	Desktop Edition	**Workstation Edition**	**Producer's Edition**
Included Features			
Tape Tools	tapectl only	Full Tape Tools	Full Tape Tools
# Tape Drives	1	2	2
Tape Library Support	No	1 @ 24 slots	1 @ 50 slots
RDX Drives	1	4	4
Doubler Mode	No	Yes	Yes
LTFS Manager & Import	Yes	Yes	Yes
Add-on Features			
50 Slot Expansion	N/A	EXP	INC
Encryption	N/A	EXP	EXP
EDL Support	N/A	N/A	EXP
Ingest/IngestPRO	N/A	EXP	EXP

Licensing ArGest Backup

Demo Mode

By default, ArGest Backup will install and run in a 30 day demo mode. There is no license information required to enable this demo period. During the demo period, all non-expansion features of ArGest Backup are available. Once the 30 day period expires, only the Restore features are available.

Entering Your License

If you have a license for ArGest Backup, you can enter it by starting ArGest Backup, Opening the "Advanced View" and selecting "Enter License" from the Application or Help menu or by typing Control-Option-Command-L (^⌥⌘L). Your ArGest Backup license will consist of 2 strings of characters - the **License Data** and the **License Key**. Enter these characters verbatim including any special characters in the fields provided.
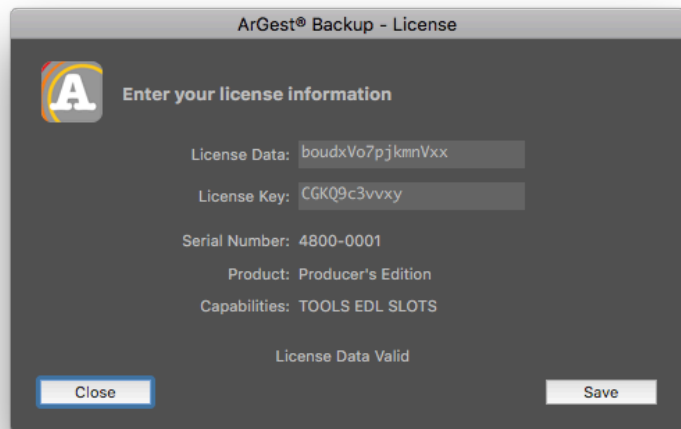
License Entry Dialog

Once entered, the license info will be verified and you may save the license.

QuickStart - Getting Started

Launch the ArGest Backup UI application on your system.

ArGest Backup supports the use of Standalone Tape drives, Tape Libraries - sometimes referred to as Autoloaders, and disk-based archives.

Once launched, you will be stepped through the Quick Configuration settings.
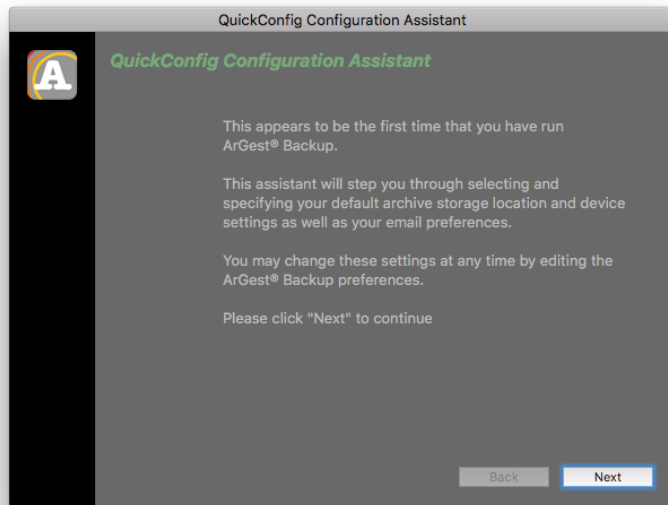
QuickConfig Dialog

Choose the Destination option that is correct for your environment.
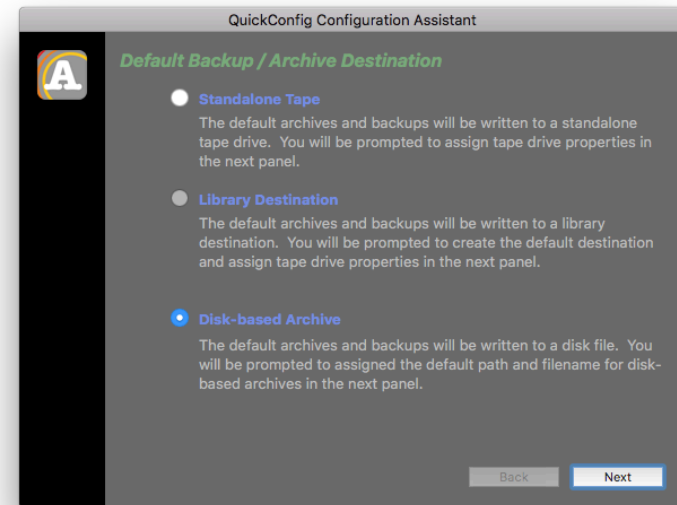
QuickConfig - Default Archive Destination

Click the "Next" button and set the Defaults for the DIsk-based file location, Standalone Tape drive parameters, or Library parameters and Destinations. These defaults will automatically be used each time you execute a backup job, but they may be changed on a job-by-job basis.

If you do not have a tape device available, your only option will be to selected a Disk-based archive. If you are working with Disk-based archives, accept the default settings and the example archive will be created in your Documents folder:

QuickConfig - Standalone Tape Settings

QuickConfig - Disk-based Archive Settings
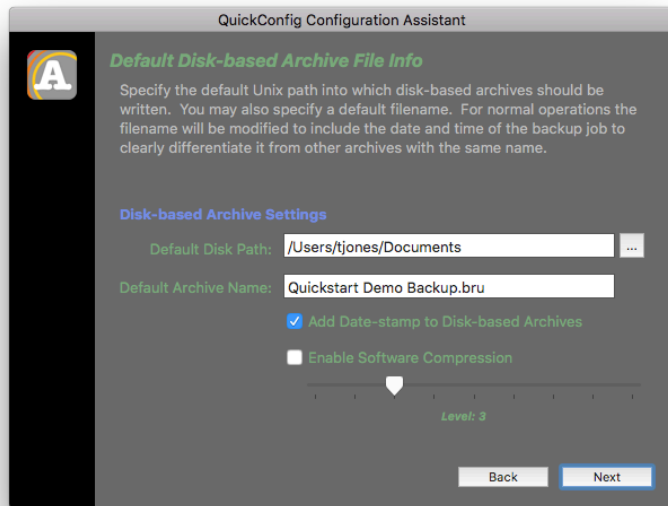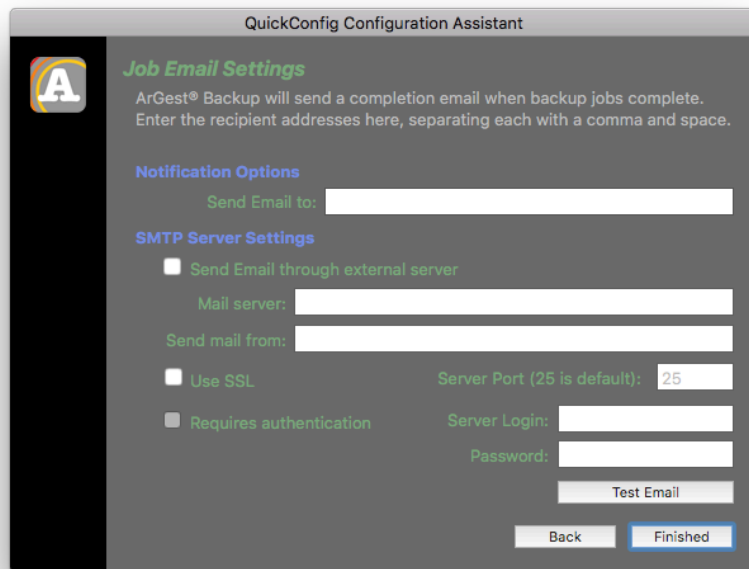
QuickConfig - Library Settings

If you are using a standalone tape drive or library, your options will allow you to set the default block size (buffer size), whether to prompt for barcodes if using a standalone drive, and whether to protect previously written tapes, overwrite previously written tapes, or append data to a previously written data.

QuickConfig - Email Settings

For the Email settings, you may fill in your appropriate information or you may leave it blank for this example.

Once the basic configuration is completed, you will be shown the Backup Wizard mode window. This is one of two paths to creating archival or backup jobs and is designed to step you through the setup and execution of a job in a manner that removes any IT-specific knowledge requirement.

Wizard Dialog

For this demo, click the "a" button to move to the next step which is to either create a new job, or select from a previously saved job definition.
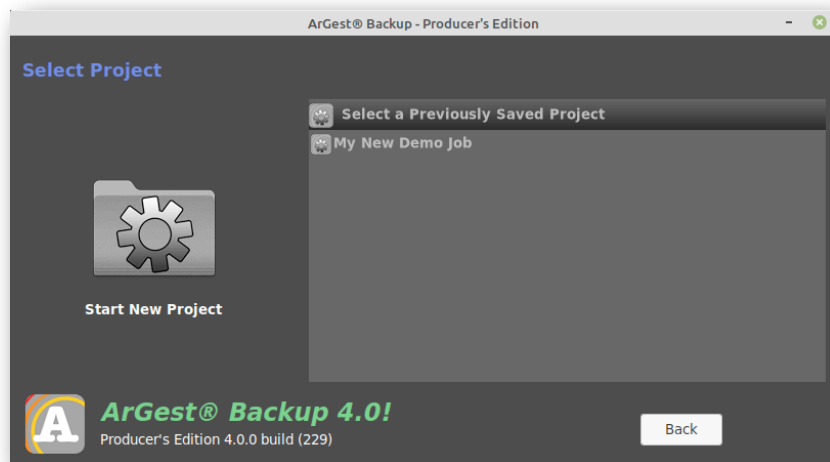
Select Project Step

On this panel, you may select from a listing of previously saved jobs, or click the "Start New Project" cog folder to start a New Project. In a new installation, you will not have any saved jobs, so click the cog folder button to start a new job.
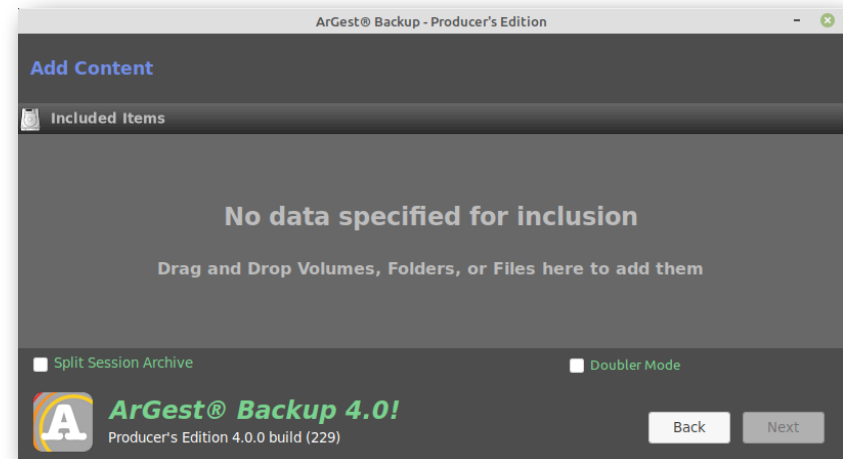
Add Content Panel

In this panel, you will define what will be included in the job, or "backed up". Adding volumes, folders, or files is as simple as dragging them from your system's file browser - Finder, Explorer, Nemo, etc. - and dropping them into the "Included Items" list.

Find a small folder containing a few hundred megabytes of data - for this example on Linux, we've chosen a folder on a shared, mounted media volume
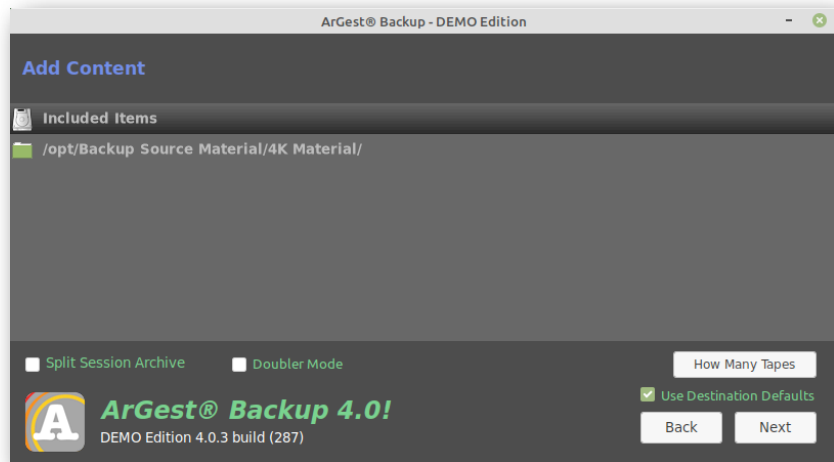
Content Added for Backup

To determine how much space or how many tapes your selected data will require, click the "How Many Tapes" button.

Click the "Next" button to move to the tape and archive naming step using the default destination that you selected in the QuickConfig steps above.
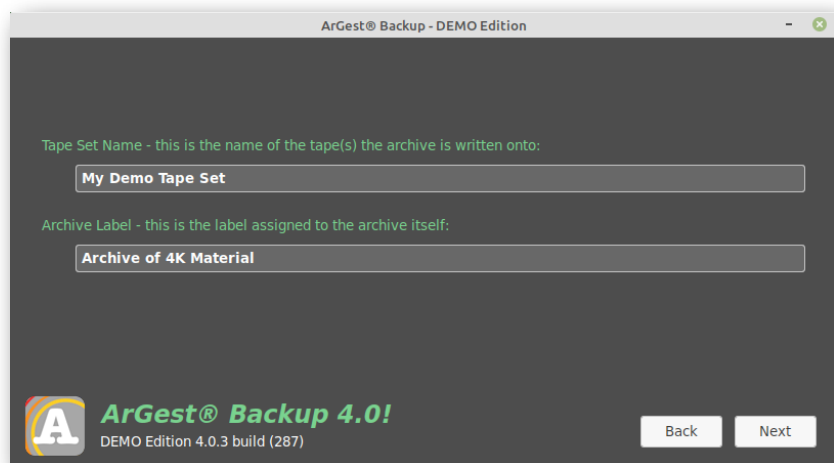
Folder Selected for Archive

Clicking the "Next" button will step to the Summary page and run an Estimate pass on the selected data. Note that the numbers shown are BRU Archive Size and will be larger than your selected data. Also, if you are using tape, the Estimate pass will report the number of tapes for your drive type and the previous generation since all LTO drives can write to tapes from the previous generation.
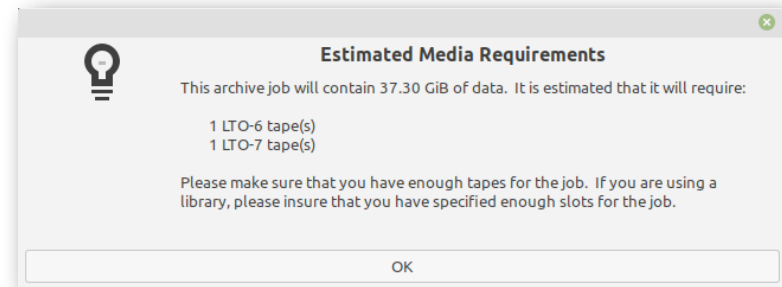
Estimate Report

Closing the Estimate dialog will display the Summary page.
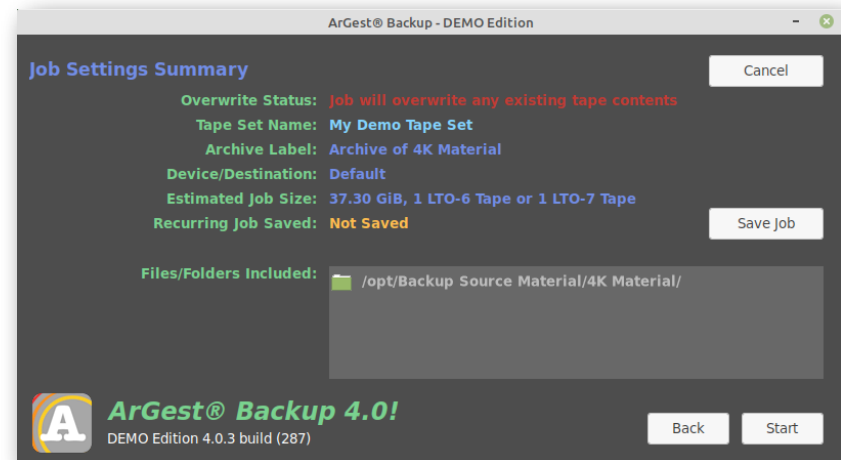
Job Summary Page

If you are satisfied with the settings as presented, simply click the "Start" button and your job will be executed.
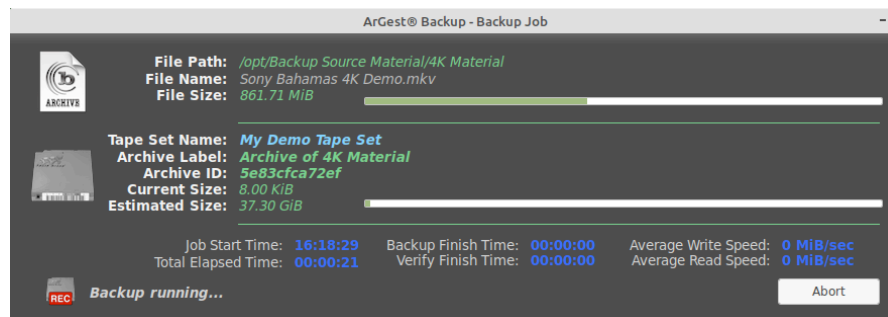
Write and Verify Dialog

Once the write and verify completes, the job Completion dialog will be displayed.
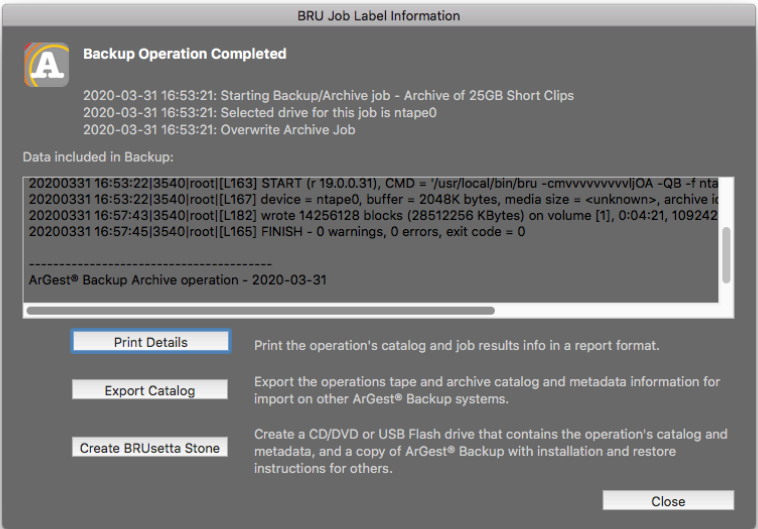
Job Completion Dialog

Click the "Close" button to return to the Wizard for now. The option options will be discussed later in this manual.

Performing the Restore

Once the job completes, click OK to close the Completion dialog and then click the "Restore" toolbar button in the toolbar of the Wizard window. This will display the Restore panel on the Advanced View window.
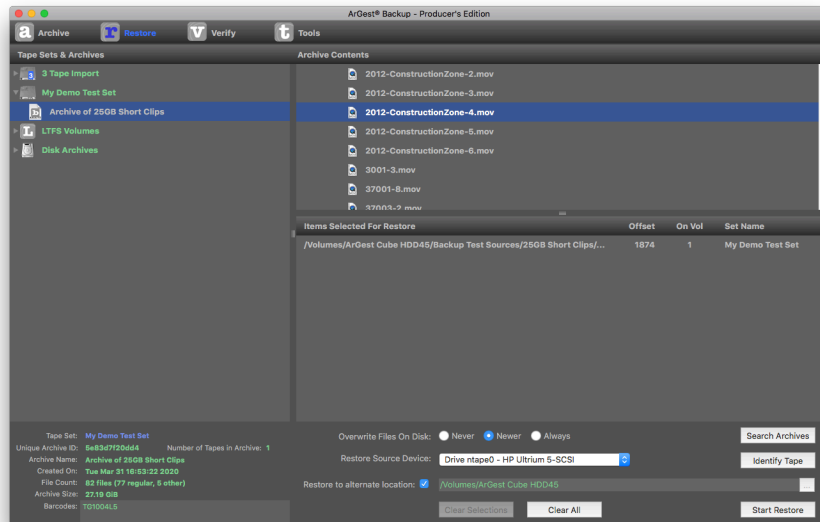
Restore Panel

Select the tape set from the listed sets (a Disk-based archive will appear under the folder "Disk Archives"). Selecting the archive will display the contents in a hierarchical list format in the upper right list. Locate the file or folder that you wish to restore and either double-click or drag and drop the selection to the lower right list.

When restoring data on an OS X system it is important to understand that even though BRU processes applications as folders, we recommend that you only restore complete top-level .app folders (many OS X Applications are really folders designed to operate as a single application) or the resulting restore could leave an application in a non-functioning state.

Next, Click the "Restore to an alternate location" checkbox and select a location from your system. We are using a mounted RAID volume in this instance.

With the alternate restore location selected, click the "Start Restore" button to restore the selected file or folder to the alternate location selected.
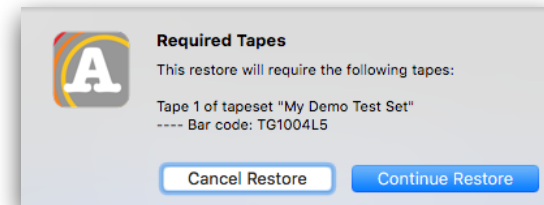
Required Tapes Dialog

ArGest Backup will prompt you with the tape(s) required along with any barcode provided if using a Tape. Once you have the tape ready, click the "Continue Restore" button. The tape will be repositioned to the starting point of the selected data and the restore will begin.
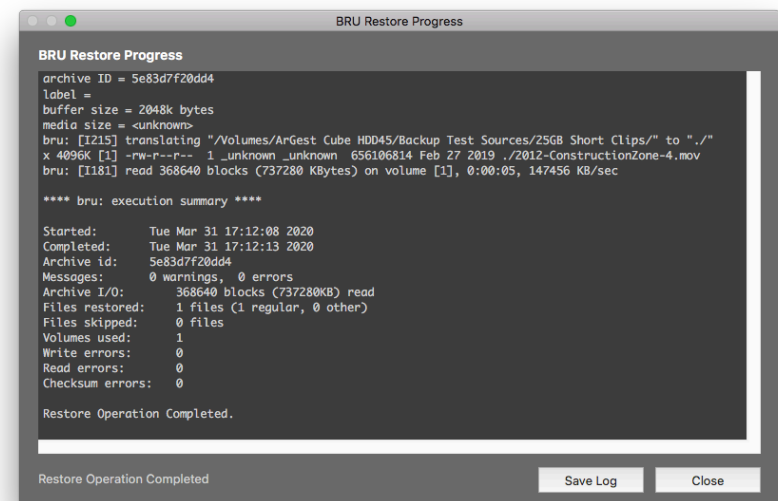
Restore Completed

If you've followed these steps, you have just performed a complete Backup, Verify, and Restore of a copy of the Messages application with ArGest Backup.

Important note for restoring to an external USB, Firewire, or non-system internal disks

When preparing an external Firewire or USB drive, the OS X default is to configure the drive to ignore permissions. If you then restore data from a ArGest Backup backup onto a drive configured in this manner, you will lose all ownership and permission information of the original files and folders. To change this setting, select the drive on your desktop and choose "Get Info" from the popup menu.
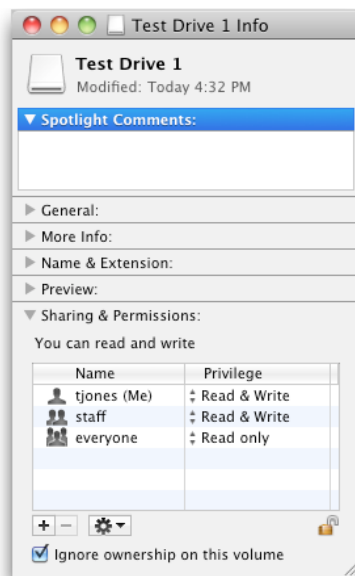
If the volume settings are locked, click the security lock icon on the lower right and enter your user password. You will now be able to uncheck the "Ignore ownership on this volume" checkbox. This change will allow you to properly restore files to that drive while retaining any ownership, permissions, EAs, and ACL assignments.

Special Considerations for Library Use

If you have a library attached to your ArGest Backup system, you will be prompted to scan the library hardware the first time ArGest Backup is run or any time that you add or remove tape hardware to the system. Since you have the option of connecting 2 tape devices to your ArGest Backup system, this operation is required so that ArGest Backup will know which physical tape drive is located within the library.

Library Scan Notice Dialog

As indicated in the dialog, please make sure that all tape drives are empty and that there is a data tape located in the first slot on your library. ArGest Backup will inform you of its progress during the scan.

Additionally, it is important that the library be placed into "Random" or "Automatic" mode for proper operation with ArGest Backup. All library bundles shipped from TOLSI Group have this set by default. For other library vendors, please check your library documentation or contact the vendor for information on how to access this setting.

During the scan, ArGest Backup loads a data tape from the first slot in the library to each library drive in turn (if more than 1). Once the drive(s) are identified, the tape is returned to the slot.

Make Sure the Library is Ready
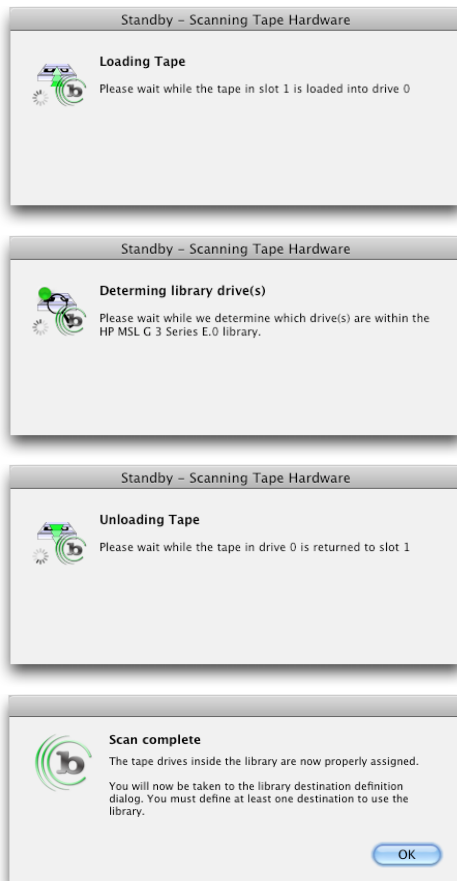
Library Scan Progress Dialogs

Once the scan is completed, you will be prompted to create your library destinations for archival use.
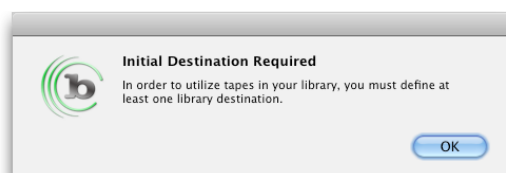
Initial Library Destination Dialog

Because ArGest Backup treats a collection of tapes in a library as a "destination", you need to assign slots to a destination before you can use your library for archive operations. A destination can consist of a single slot or all of the slots within your library. The only limitation is that ArGest Backup destinations *MUST* be made of contiguous slots and should not overlap.

Additionally, when dealing with the tapes within a destination, you *MUST* handle all tapes in the destination as a single set and not handle the tapes as separate entities. Therefore, *when dealing with the tapes in a destination, you must remove and replace all tapes in the assigned slots to change or reuse the destination. You should never remove "some" of the tapes in a destination*.

If you need to modify the slots assigned to a destination after it has been used, it is also very important that you change the tape(s) in the destination as the slot handling will change and confusion over the media used could result in tape handling issues for the humans involved.

We also do not recommend using perpetual appending of incremental data to a library destination as you will eventually end up needing to replace the media within the destination. The current version of ArGest Backup will allow you to append data to a tapeset in a destination as long as the current operations have not exceeded the capacity of the original media within the destination. Once operations to a destination's tapeset have exceeded the capacity of the tapes that are originally loaded, ArGest Backup will no longer allow you to append to members of that tapeset. If you require this type of operation, our BRU Server solution is designed specifically to support it.

To put it simply, once you define a ArGest Backup library destination, treat it as one tape. If you must change the destination's slot assignments, remove any existing media and replace them with fresh tapes. Do not develop a backup or archival scenario that assumes you will be able to append to a destination's tapes forever.

!! Removing a portion the tapes from a destination's assigned slots will confuse ArGest Backup's tape handling logic and potentially result in unexpected overwriting of data on tapes within the destination.

Library Destination Manager

When defining a destination, Give consideration to the amount of data that will be written to the destination as a whole. For example, if you are using an LTO-4 device, the tapes are rated for 800GB per tape. If you determine that you will be backing up 2TB of data, you should allocate 3 tapes to the destination. If you have the time, you can get a closer determination of the number of tapes by running the full backup of the data to a destination consisting of 3 tapes. ArGest Backup will tell you how many tapes were used for the actual backup.

Since there is no way to estimate the level of compression for a given data selection, it is best to only use the native capacity when determining the number of slots to assign to a destination. For information on the native capacity of each tape type, please refer to the performance table previous chapter of this manual.

<div style="text-align: right">

3
Configuration and Preferences
</div>

QuickConfig

You are shown the QuickConfig dialog the first time you run BRU PE or any time you click the QuickConfig button on the Wizard toolbar. The QuickConfig dialog presents you with the minimum number of options for configuring and running archival operations with BRU PE. Once you make the appropriate selections in QuickConfig, you will only need to modify them if you are changing the default archive destination or you wish to fine tune I/O performance.

The default archive destination allows you to choose Where you want the archives to be written - Standalone Tape, a Library Destination, or a Disk-based archive.
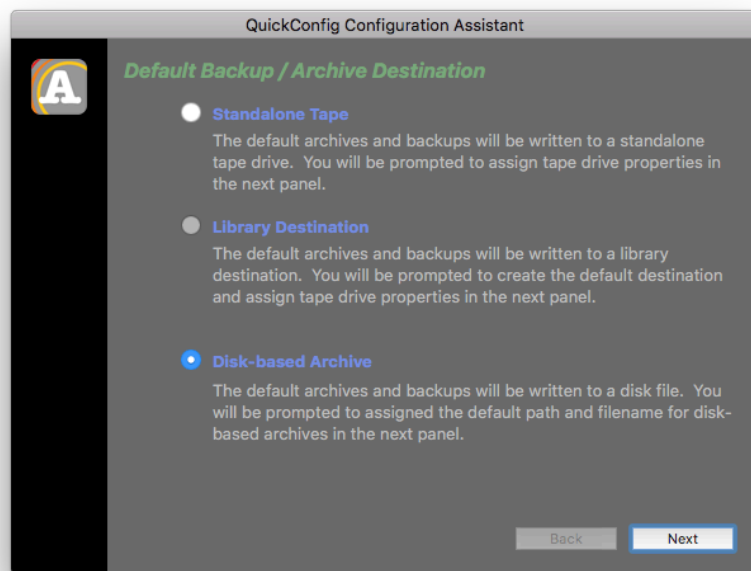
QuickConfig - Default Archive Destination

Standalone Tape Drive

If you have a standalone tape drive (a tape drive not installed in a library) select Standalone Tape and click "Next".
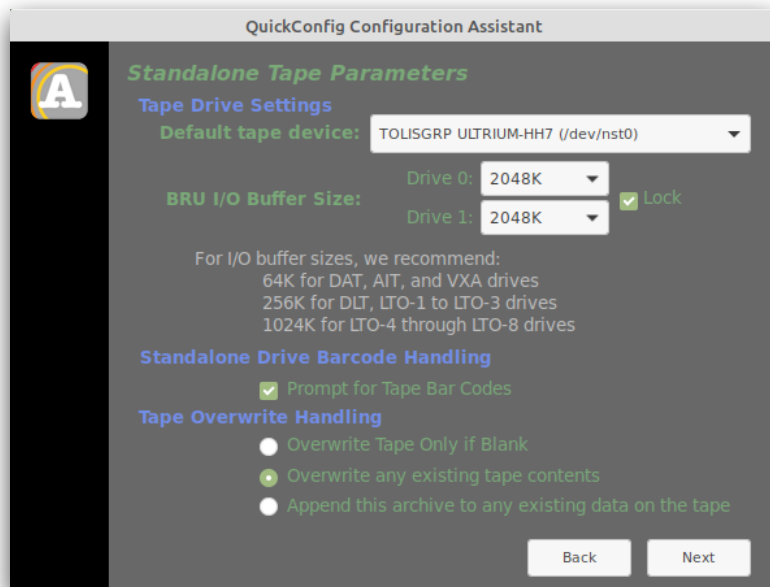
QuickConfig - Standalone Tape Settings

The basic options are Buffer size, Prompt for manual barcode entry, and default overwrite state.

The overwrite option must be noted as it is automatically applied in Wizard Mode. However, ArGest Backup will warn you if you are going to overwrite a known tape.

When choosing a buffer size, you should consider the type(s) of data that you will be archiving. If you are constantly archive small files (email, documents and spreadsheets), a smaller buffer size will work best. However, if you are archiving music, movies, or other large media files (4MB+), then choosing a larger buffer size can speed up your archival operations.

Buffer Size Limitations

When working with various tape drives, there are some limitation when assigning buffer sizes. The following tape outlines these limitations:

Device	Max Buffer	Notes
DAT	512K	While you can use greater than 512K, there is no further performance gain.
VXA-1/2	128K	This is a limitation of the drive technology.
VXA-320	256K	The current firmware version errors on larger buffer sizes
Non-SAS DLT Variants	1024K	
SAS DLT Variants	2048K 1024K	2048K is supported with the ATTO ExpressSAS HBAs. If using the LSI SAS HBA, this is limited to 1024K
LTO-1	1024K	This is a limitation of the drive technology.
SCSI LTO-2 through 4	1024K	Larger buffer sizes demonstrated no addition performance gains
SAS or Fibre LTO-3 through 8	2048K	Larger buffer sizes demonstrated no addition performance gains

Buffer Size Limitations

When possible, ArGest Backup automatically limits the buffer size selection options.

Library Destination

ArGest Backup handles libraries as collections of slots that we refer to as Destinations. When you select to use a Library Destination as your default archive device, your first step is to define the slots that will be used for the archival operations. This allows you to subdivide your library into sections for better archival management.
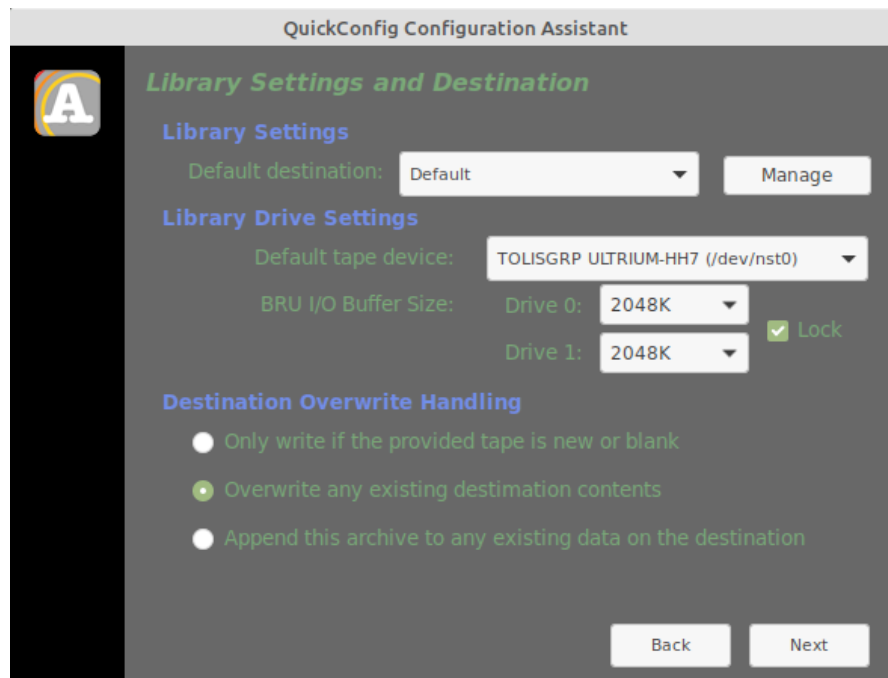
QuickConfig - Library Settings

By default, there is a library destination created for you named "Default". This entry uses all of the slots in your library as a single destination. If that is all you need you may skip the next step and select "Default" from the Default destination popup menu.

The settings for buffer size and overwrite are the same as discussed in the Standalone Tape Drive section above.

If you wish to segregate your library into multiple destination, click the "Manage" button.
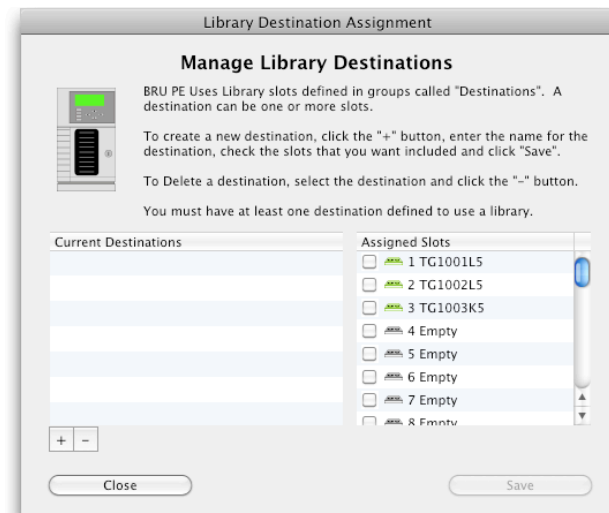
Library Destination Manager

To create a destination, check the slots that are to be assigned to the new destination. Be aware that destinations should not overlap one another as this could cause an unexpected overwrite of part of another destination's archives. When you are satisfied, click the "Save Destination" button.

Continue until you have created all of the destinations that you will need. Click the "Close" button and you will be returned to the QuickConfig dialog. Select the destination that you just made as the default for your archival operations.

For more important information about using ArGest Backup and Library Destinations, see

Disk-based Archives
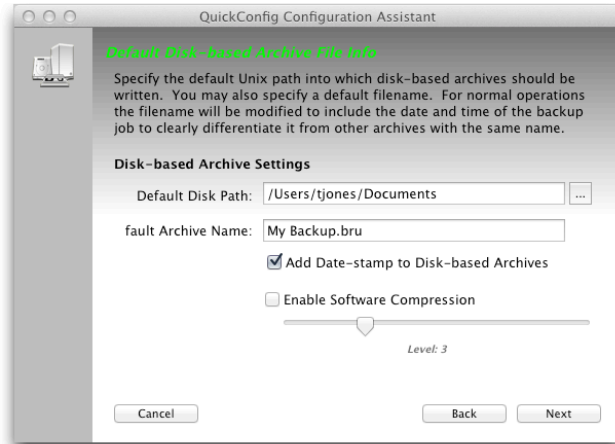
QuickConfig - Disk-based Archive Settings

If you will be writing archives onto Disk, you may specify the default path and archive filename, if the files created should have the archive time and date added to the name, and whether you should use software compression.

While the path and name of the archive files are self explanatory, adding the Date-stamp requires explanation. When ArGest Backup creates files on disk, if a newer file has the same name as an existing file, the older file will be overwritten. Therefore, without modifying the new archive's filename, you will inadvertently overwrite the existing archive with the same name. By adding the new archive's date and time to the archive's filename, you remove this potential. A date-stamped filename will look like this:

```
My Backup-20090313.204059.bru
```

This file was created on Mar 13, 2009 at 20:40:59. Since we track the time to the second, you will never run into a situation where one file will overwrite another unexpectedly.

Email Settings
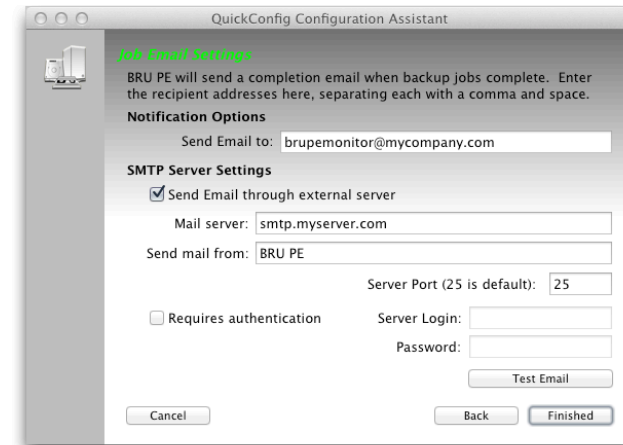
QuickConfig - Email Settings

ArGest Backup will send completion emails if you configure your email settings. This panel allows you to set the recipients and any mail transfer settings that are required.

The "Send Email to:" field can be a single entry, or a list of recipients separated by commas.

If your system must send email through an external server - such as Apple's Me.Com or Google's Gmail, Check the checkbox and enter your details. Note that the "Send Mail From" and "Server Port" entries may have specific requirements based on your mail account on the remote server.

If you must authenticate to send mail, enter your user name and password credentials here. While your password is stored within the ArGest Backup preferences, it is encrypted for security.

If you wish to test the settings, clicking the "Test Email" button will send a test message using the settings entered.

Once you are satisfied with these settings, click the "Finished" button to save the settings and exit QuickConfig.

As mentioned in the QuickConfig section of Chapter 2, once you've completed these steps, you don't need to revisit the configuration

properties unless the hardware or email settings for your system change.

Advanced Preferences

If you wish to modify more of the ArGest Backup preferences than those available in the QuickConfig option, you can use the standard Preferences dialog to gain access to all of the settings available.

These preferences include Backup, Restore, Devices, Email, and General settings.

Backup Preferences
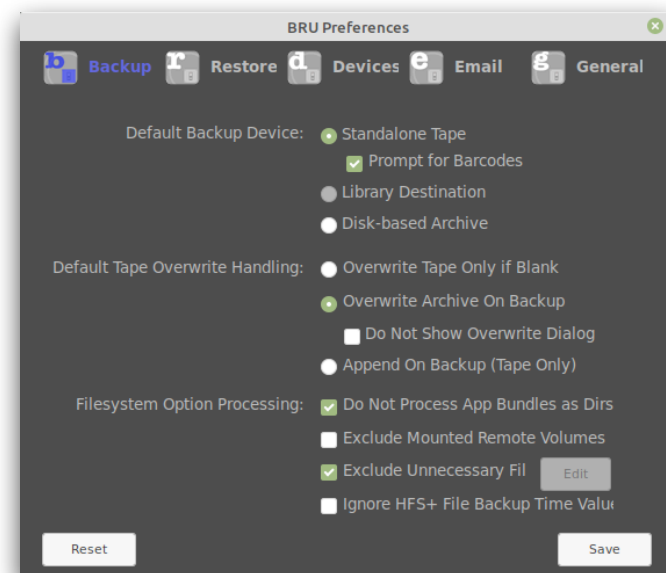
Preferences Window - Backup Preferences

The Backup Preference settings allow you to choose whether you wish to use a standalone tape drive, a library destination, of a disk-based archive as your default archive destination.

The overwrite handling sets a default value for whether a tape or library destination will be overwritten when an archive is created. This option does not apply to disk-based archives since your only options for disk archives is to either create a new archive or overwrite an archive with the same name.

The Filesystem options allow you to utilize and create an exclusion list that will exclude files by name, path, or wildcard. You can also choose whether Application bundles will be processed in the UI as a single file or will be exposed as a folder.

When performing backups, you can instruct ArGest Backup to ignore certain files, paths, or patterns. To enable this option, check the "Exclude unnecessary files" checkbox. You may edit the contents of the bruxpat (the BRU eXclude PATterns) file by clicking the "Edit" button. The bruxpat edit dialog will be displayed.
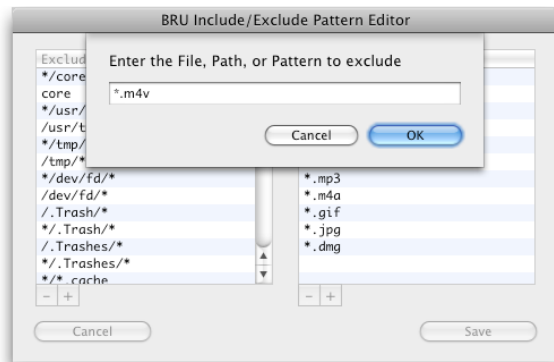
The bruxpat Edit Dialog

To add an entry to the exclude or non-compress lists, use the "+" button. Entries created with the editor will be defined as standard shell-pattern excludes. To remove an entry, select the entry and click the "-" button. For more information on the bruxpat file, see Appendix I later in this guide.

For detailed information on the bruxpat file and its contents, refer to Appendix I later in this document.

Restore Preferences
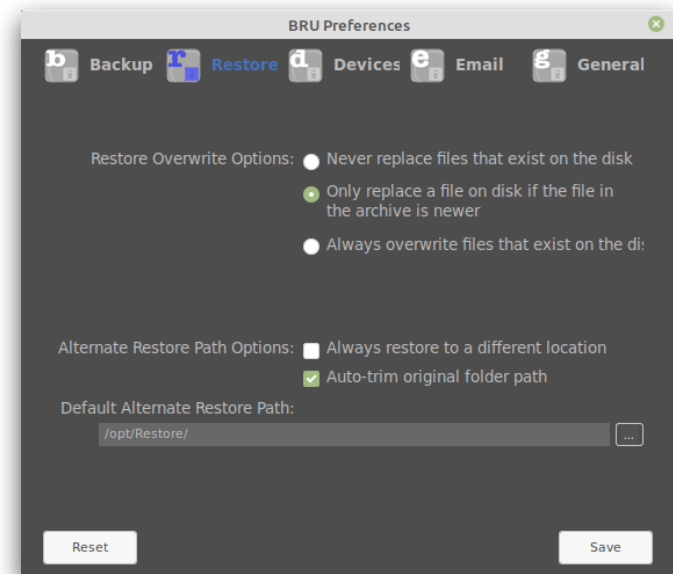
Preferences Window - Restore Preferences

The Restore Preferences options allow you to define how restores are performed by default.

Normally, ArGest Backup will only overwrite a file on disk if the same file in the archive is newer than the file on the disk. You may override this functionality by selecting either "Never replace files that exist on the disk" or "Always replace files that exist on the disk."

When restoring data on an OS X system it is important to understand that even though BRU processes applications as folders, we recommend that you only restore complete top-level .app folders (many OS X Applications are really folders designed to operate as a single application) or the resulting restore could leave an application in a non-functioning state.

Device Preferences
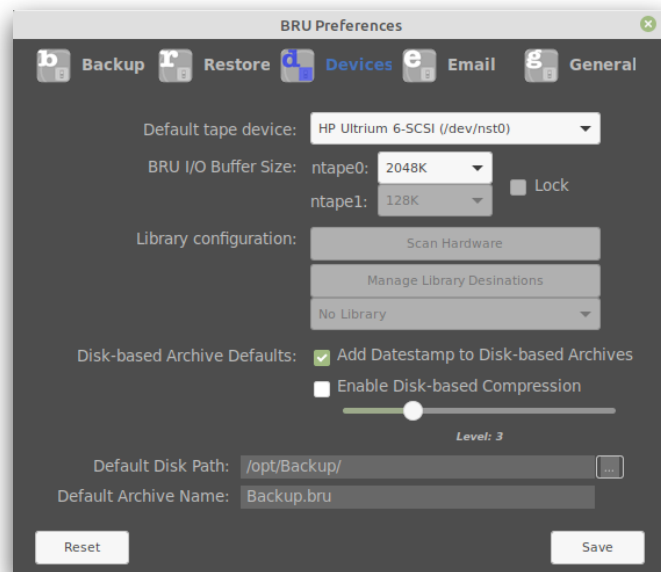
Preferences Window - Device Preferences

The Device Preferences allow you to choose the default tape device or library destination, as well as the tape block size or the default disk-file path and archive file name as well as the date stamp rules and whether to use software compression when writing to disk.

The blocksize can be used to improve device I/O speed when writing to a tape drive. Modern, high-capacity drives have large I/O input buffers and using a larger I/O blocksize can improve the speed of archive operations. Refer to the table of devices in the QuickConfig section above for the best blocksize settings for your device type and connection.

The software compression only applies to archives written to disk as the compression mechanism is automatic when writing to tape.

Email Preferences
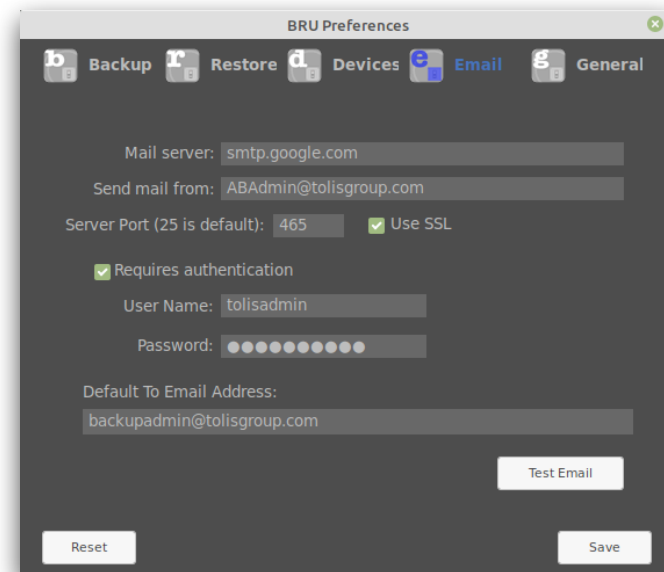
Preferences Window - Email Preferences

You should fill in the email settings depending on your mail transfer requirements. If you are sending through an external host, please get the proper port and login information to use.

Once you have entered your settings, you may send a test email by clicking the "Test Email" button to make certain that the settings are correct before saving them.

General Query Preferences
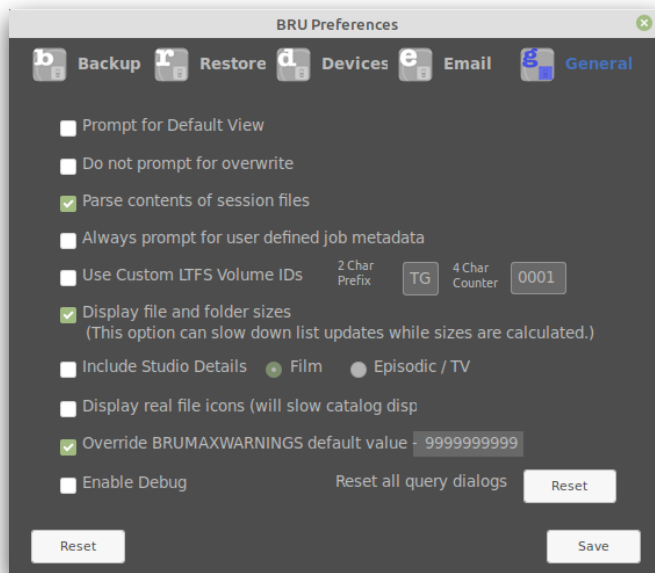
Preferences Window - General Query Preferences

The final preferences panel is the "General Query" settings panel.

Prompt for Default View

Normally, ArGest Backup only prompts to select the default view -
Wizard or Advanced Mode - the first time it is run. Checking this
checkbox will cause ArGest Backup to redisplay the selection
dialog the next time that it is run. This is useful if you are
configuring a system for someone else and wish to have that
choice presented the next time they start ArGest Backup.

Wizard as default mode

This checkbox will change the default display mode the next time ArGest Backup is run. It does not cause the selection dialog to be displayed.

Do not prompt for overwrite

Normally, ArGest Backup will always prompt you before overwriting a non-blank tape - one that contains data (BRU or otherwise). Checking this option will automatically overwrite any tape provided.

This option has no impact on disk-based archive operations.

Do not ask to name unnamed tapes

If you do not wish to be prompted to provide a name for unnamed tape sets in the Restore panel, check this option.

Parse contents of session files

If you are working with sessions from supported DAW and NLE software such as Final Cut Pro or Logic, selecting this option will instruct ArGest Backup to parse the session file to find the actual files associated with the session. With this mode enabled, ArGest Backup will automatically track the clips or stem files in a session to include them in a session-specific archive operation.

Deselecting this option will cause ArGest Backup to treat a recognized session file as a plain file and the associated clips and other files will not be included automatically.

Use custom LTFS volume IDs

If you are using LTFS support, each volume that you format in ArGest Backup's LTFS Manager will be assigned a custom volume ID. This ID is a 6 digit value that will consist of a prefix of 2 alphanumeric characters and a counter value of 4 numeric characters. The 4 numeric values will be incremented for each LTFS volume that you format on a specific system. If you are formatting LTFS volumes on multiple systems, you should work out a scheme by which the values applied do not conflict as this volume ID is the unique value used to differentiate tapes. The default values for ArGest Backup is "TG0001", therefore your tapes will be formatted as TG0001, TG0002, TG0003, etc. unless you

change these values. This setting allows for up to 9,999 unique LTFS volume IDs.

Display file and folder sizes

Enabling this option will display the sizes for files and folder in the Advanced Mode Backup panel. This option can seriously slow down the displayed information when expanding folders and volumes and the contents are examined to provide the total folder or volume content size information.

Always Prompt for user-defined Job metadata

The operator may provide metadata that applies to the actual ArGest Backup job. This metadata consists of 4 free-form text fields that can be searched in the restore panel to locate a specific archive based on that metadata.

Use custom LTFS volume IDs

If you are creating LTFS tapes using ArGest Backup's LTFS Manager, you may change the default serial numbers for your needs using these two fields.

Include studio detail

This option prompts the operator for project details retailed to either film or episodic productions. These options are aimed at productions that will be vaulted and utilize the ArGest Backup archive tape set info QR Code generator.

Enable debug

This option will turn on ArGest Backup's debug mode and open a special Debug output window. You should only run with debug enabled if instructed buy TOLIS Group support personnel.

Backing Up the ArGest Backup Operating Environment

While it is easy to recover your archive catalogs from your tapes using the Import Tool, this will not protect your other ArGest Backup preference settings and any saved definitions in the event that you suffer from a system failure.

You can manually create a backup of the environment from the "***Advanced View***" -> "***Tools***" panel. On The "***Environment***" tab, you will find pushbuttons that allow you to manually backup and restore the ArGest Backup environment interactively.
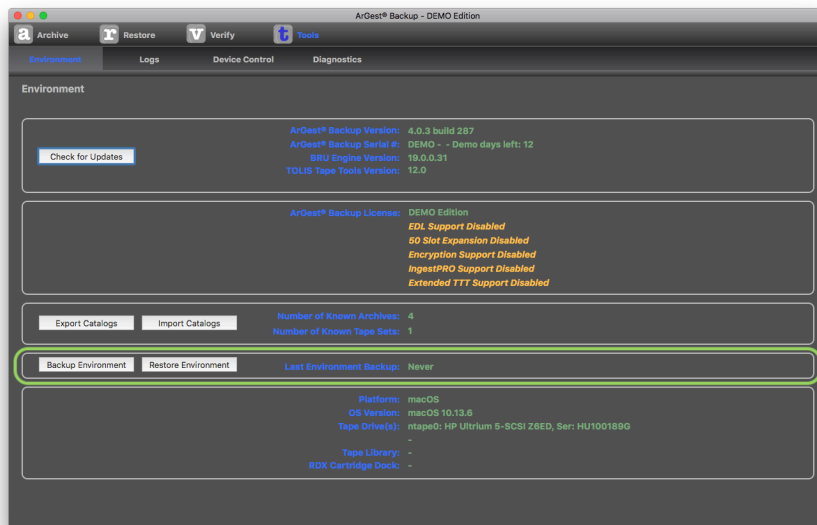
Environment Backup and Restore Options

The "Backup" option will allow you to choose a location on your system to save the resulting ".bru" file. Good locations are USB thumb drives, or a remote location such as iCloud, OneDrive, GDrive, or DropBox. The idea being to have a copy of the ArGest Backup environment in a location that is not on your system for recovery purposes.

The Environment Backup is not the same as a Catalog Export since the Catalog Export ***ONLY*** contains your archive information.

If you are creating repeating jobs, jobs that will be run as full and incremental or differential, jobs that will run in unattended scheduled mode, or backup jobs that require special options beyond ArGest Backup's default settings, Advanced Mode is your best option for the default mode of operation.

__4__
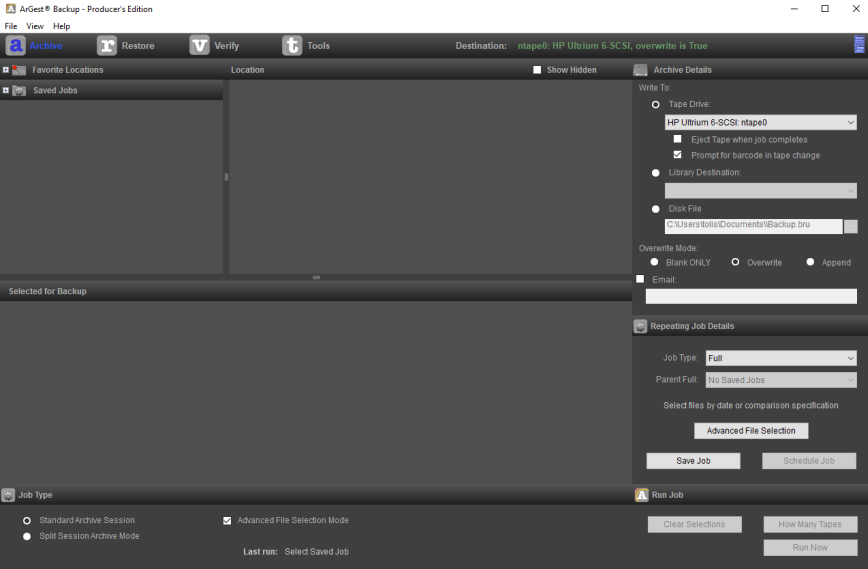
ed View

kup or

st time that

king the

n the Window

in which you
run your backup or archival operations most often.
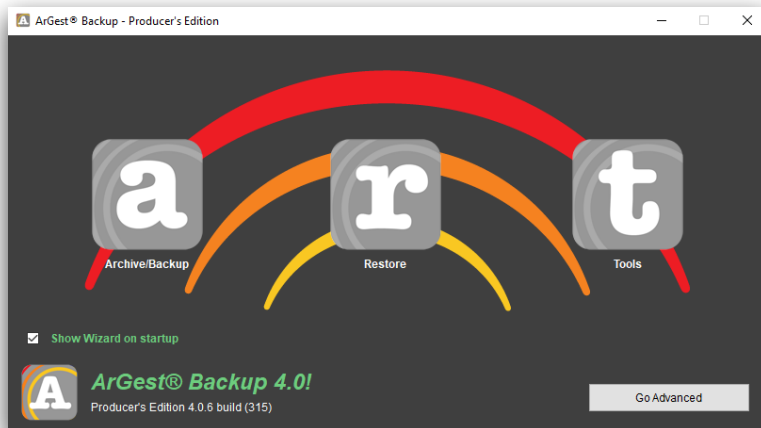
Wizard Mode View

If you are performing mostly one-shot operations to tape or disk, or using the Doubler Mode of operation, Wizard Mode is probably your best default selection. Wizard Mode is designed to provide the simplest interface for backup operations that you run based on a single job requirement or to quickly kick off a backup that can be run with ArGest Backup's defaults or doesn't need a lot of special options. The idea is a simple one - drag your files or folders from Finder or your desktop to the listbox, load a tape and click the "Start Archive" button.

Advanced View Mode

5
Using Wizard View

Overview

Wizard View Window

The Wizard View window consists of three buttons -

- a = Archive Operation
- r = Restore Operation
- t = Tools and Diagnostics

Additionally, you may disable Wizard mode to start in Advanced Mode each launch, or jump from Wizard mode to Advanced mode by clocking the "Go Advanced" button

Wizard Mode was designed to provide a very easy to use, drag and drop, operation that requires no knowledge of your filesystems or tape device operation. As demonstrated in the QuickStart chapter, you simply drag Volumes, Folder, or files from the Finder to the target listbox in Wizard View mode, insert your tape, and click the "Start" button.

Archive mode – Select Project
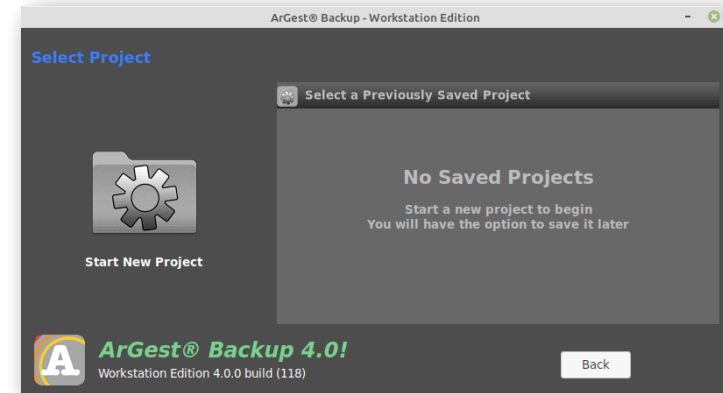
Select Project Panel

The Select Project panel will allow you to either "Start a New Project" or select a previously saved project from the list.

Archive Mode – Adding Selected Data
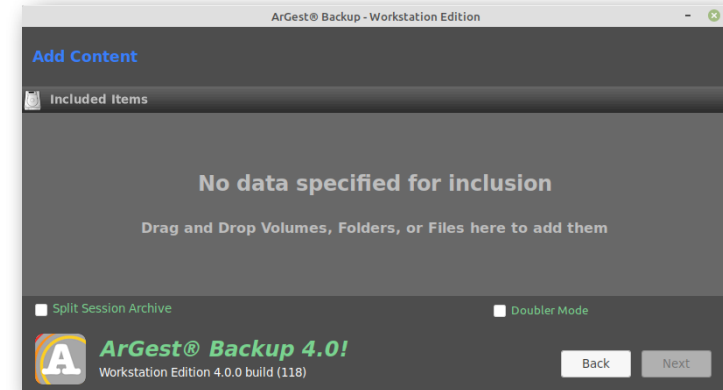
Add Content Panel - Empty

Adding File Selections
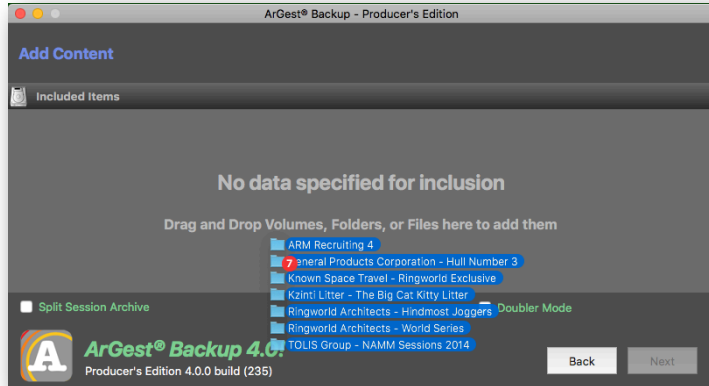
Adding Selections for Archival via Drag and Drop

Adding volumes, folders, or files to a Wizard operation is performed by dragging and dropping your selection from your desktop or the Finder to the Wizard list box.

To remove an entry, select it in the list and press the Delete or Backspace key.
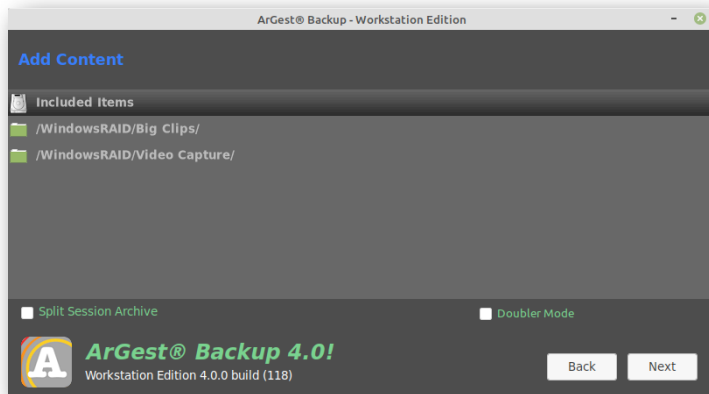
Add Content Panel Selections Added

Once you have added the volumes, folders, or files that you wish to include, click the "Next" button.

Archive Mode – Setting the Destination
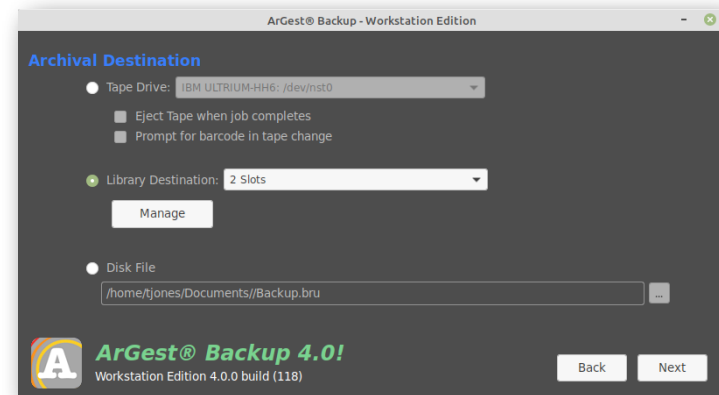
Select Destination Panel

By default, the destination that you selected in the QuickConfig steps will be automatically selected here. However, you may change that setting on this panel. If you are satisfied with the assigned destination, click the "Next" button.

Archive Mode – Job Details
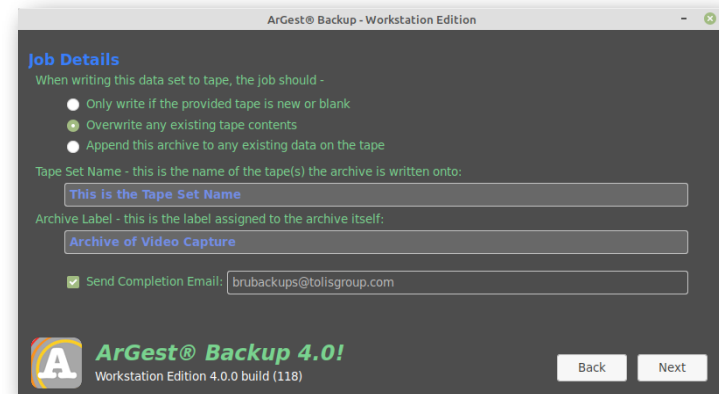
Job Details Panel

As with the Destination panel, you may leave the default information, or change it to suit this specific job. Once you are satisfied with the overwrite and naming settings, click the "Next" button to continue

Archive Job – Media Estimate

Wizard Job Media Estimate

The estimate result will indicate how many tapes the job will require based upon 1:1 non-compressed data. If your data is compressible, these values may be greater than the final result. The reason that we list tape count for both your drive type and the next generation back is because you can mix tapes or use the previous generation media to perform the task.

Archive Mode – Job Settings Summary

Wizard Ready To Write

At this point, the job is ready to launch. Or, by clicking the "Save Job" button, the current settings will be saved for reloading at a later time to duplicate this operation.
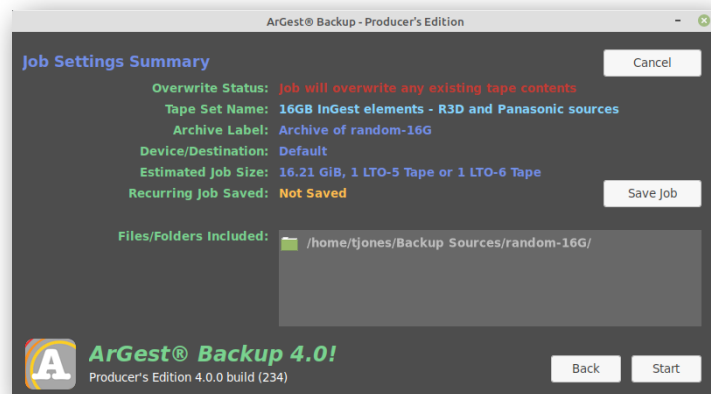
Split Session Archive Mode

With modern tape technologies now holding 2.5TB, 6TB, or 12TB of data on a single tape, it is often more economical to store more than a single project on a single tape. ArGest Backup's "Split Session Mode" is designed to make it easy for the creation of a single tape that contains segmented archives containing related files in each set.

As an example, if you are a post house supporting the work of multiple clients, you can store the archival copies of your clients work in separate archives on a single LTO-5 tape. By separating the clients' content into separate physical archives on a tape, recovery of a single client's data is easier to manage while not requiring separate tapes to archive the data.

Separate Sessions - Restore Client Data
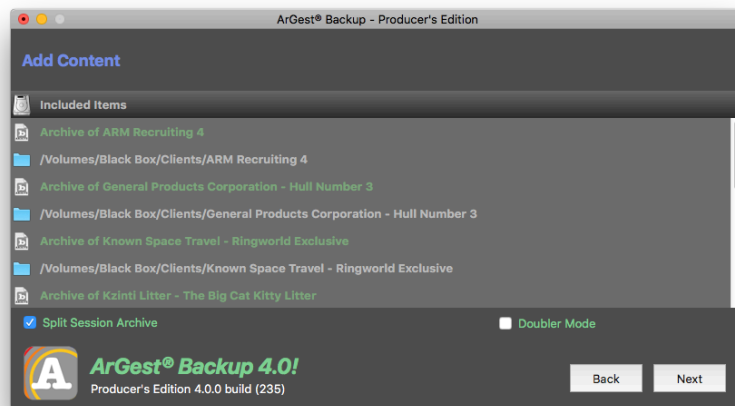
Wizard Mode - Separate Sessions

During the actual archival operation, the job status will display the progress of all jobs.

Once a Separate Sessions tape is created, the archives will be displayed as separate entities in the restore panel as shown below. This makes retrieving data for a specific client or client project easier since each specified section is a separate physical archive entity on the tape you don't have to search through all of the data on a tape to retrieve one specific set of client data.

Estimate Media

After you have selected the data for archival or backup, clicking the "**Estimate Media**" button will run a BRU estimate of the data selected and provide an estimate of the number of tape volumes required for the job. The values presented are based on the media's native, non-compressed storage capacity. Additionally, the estimated number of tapes will only be displayed if you are using an LTO-4 or LTO-5 drive. In other situations, the estimated total of BRU data to be written will be displayed and you will need to determine the number of tapes based upon your drive and media capacities. Since it is not possible for BRU to estimate the level of compression that a given data set will achieve on tape, we recommend that you only use the native capacity of your media when determining the number of tapes that will be required.

Media Requirements Estimate Dialog

In the case of a Doubler mode archival operation, you will be provided with the total number of tapes for each job and the jobs combined.

Tape Doubler Archive Mode

If you have 2 tape drives attached to your system, ArGest Backup supports the use of both drives at the same time to create 2 copies of the same data at once. This is referred to as "Doubler Mode" since you are basically doubling your archive without requiring you to run two separate archival operations. This is designed to allow you to easily create both onsite and offsite archival copies or local and client-deliverable copies of your selected data.

To use Doubler Mode, Select the data to be included in the archives. Click the "*Doubler Mode*" check box below the selected paths listbox to enable Doubler Mode. When you click the "Start" button, you will be prompted to insert a new tape into both drives. Note that the "*Doubler Mode*" check box is disabled if you do not have 2 drives, or if you have one standalone drive and the second drive in a library.

After you start the archival operation, ArGest Backup will perform an estimate pass on the selected data. Once the estimate is completed, you will be informed of the estimated size of the job and the number of LTO-4 and LTO-5 tapes for each of the Doubler Mode operations.
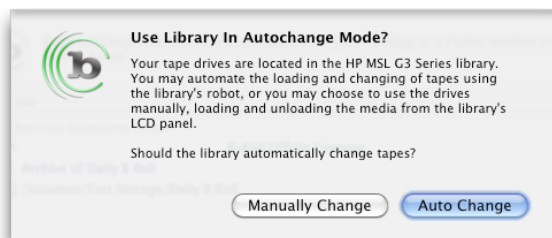
Allow the Library to Manager Tape Changes In Doubler Mode

If you have a supported two drive library such as the TOLIS Group TGL2240 or the HP MSL-2024, ArGest Backup 2.3.0 supports using the library in Doubler Mode. If both of your tape drives are within the library, ArGest Backup will automatically split the library in half - assigning the lower slots to Doubler Job 1 and Higher Slots to Doubler Job 2. If you prefer, you can change the slot assignments before the job starts.
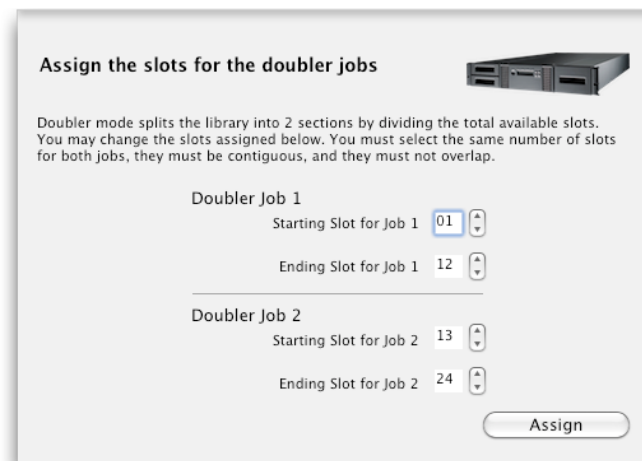
Library Slot Assignment for Doubler Jobs

The only restrictions on slot assignment is that the slots used for Job 1 must be lower slot values than the slots used for Job 2 and the slots assigned for each job must be contiguous and not overlap.

If you select a range of slots for each job when you run a Doubler Mode archival job to a library, ArGest Backup will pause the start of the second job to reduce the impact of tape changes between the two jobs conflicting for library robot access.

Because Doubler Mode is designed for archival purposes, ArGest Backup does not support appending data to tape so any tapes used will be overwritten - any existing data irrecoverably erased. Additionally, you should always use new media for this purpose to insure the highest life expectancy for tapes created for archival purposes.

When you start a Doubler Mode archival operation, you will be prompted to enter 2 tape volume names. These names are defaulted to the name that you gave the tape when you dropped your data onto the list box the the words "- Copy 1" and "- Copy 2" appended. To provide for a more useful definition when parsing the tapes for restore, or providing the tapes to a client as part of their Client Deliverables package, we recommend that you choose names that will be more representative of what each tape set is for.

For example:

My Super Movie - Day 11, B-Roll On Set Use Only

My Super Movie - Day 11, B-Roll, Offsite XDCAM 3 and 4 Raw

As you can see, the names for the tapes are descriptive and easily recognized both in the day to day operations, or 2 years after the film has completed its 42 week box office run. These names can be any text up to 128 characters.

Note about I/O Performance in Doubler Mode

Doubler Mode Tape Performance Notice

In addition to the native speed of your tape drive, the speed at which your system can write your selected data to tape is limited by four things -

- Your system's buss type (PCIe-2: all Macs and older PCs, PCIe-3: newer PCs)
- The number of PCIe lanes available for your HBA (1, 4, 8, 16: slot dependent)
- The contiguous read speed of your source disk volume
- Your network speed and latency if your storage is on a shared storage device

For an LTO-5 or LTO-6 drive, most systems - including Mac's using Thunderbolt - are capable of streaming 2 tape drives assuming a fast source volume.

For LTO-7 and LTO-8 drives, you will run into a bottleneck with the system PCIe-2 buss and the fact that most systems only provide 4 lanes (PCIe-2 x4) for I/O to the SAS devices.

☑ Do not show in the future   I Understand

Doubler Mode Performance Notes

When selecting Doubler Mode operation for the first time, the Performance Notice dialog is displayed. It is important that you understand the factors that are involved with tape performance when running two, simultaneous jobs containing the same data. This dialog explains that system considerations that you should be aware of.

<div align="right">

_6_
Using Advanced View

</div>

Overview
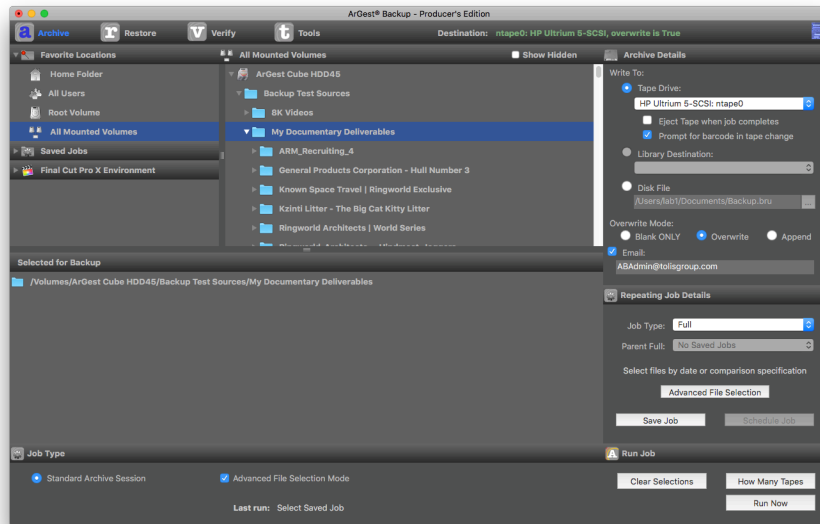
Advanced View Backup Window

ArGest Backup's Advanced View mode offers access to a much larger set of BRU's processing capabilities. Where ArGest Backup's Wizard View mode is designed to mask the majority of BRU's file and archive processing capabilities and provide a much simpler mechanism for performing simple archival operations, ArGest Backup's Advanced View mode is just the opposite in that it exposes a much larger set of BRU's capabilities.

In the Advanced View mode, the user may select data for backup and then create job definitions that are saved and either run manually or scheduled for repetitive execution and predefined times. The user may also create, save and schedule hierarchies of Full, Incremental, and Differential operations that use the history of previous backup jobs to select and backup file and volume data based on specific definitions.

Advanced View Toolbar

The toolbar allows the operator to choose the current mode of operation - Archive, Restore, Verify, Tools, and to display the backup settings pane.

Selecting Files

Volume/Path/File selection is made in the upper list box. To select entries, you may double-click an entry or drag the entry from the upper list box to the lower list box. In either instance, the selected entries in the upper list box will be added to the lower list box. Only those paths or files listed in the lower list box will be included in the backup that is run.

By default, the upper listbox displays the user's home folder. Clicking the buttons to the left of the list will change that listbox to display "All Users", the "Root" drive, or the entire "System" (all mounted volumes).

Starting with ArGest Backup 3.0, the Advanced file selection lists allow you to view and manage your default folder paths and saved jobs more easily.

The Locations folder will display four default locations - Your user's Home folder, The top level Users folder, The "root", or system drive folder, and the mounted Volumes folder.
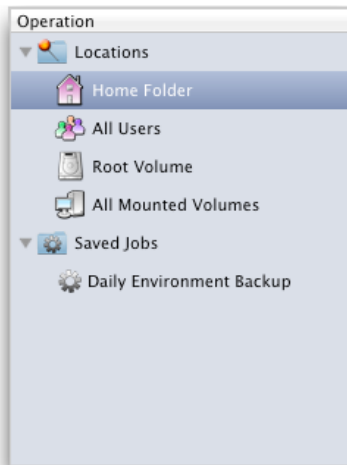
Locations and Saved Jobs

However, if you have special folders that you regularly access for archival operations, you can add them to the default locations by dragging them form Finder to the "Locations" line and then will be added to the list of locations that are displayed when you expand that entry.
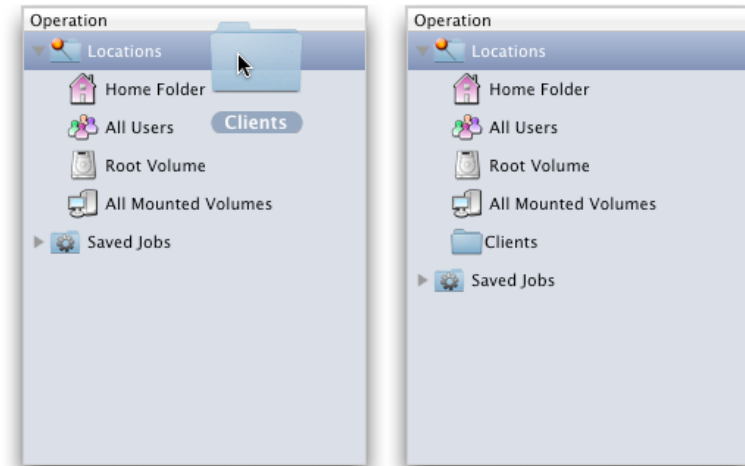
Drag and Drop

After Drop
Adding a New Location

These options allow you to limit or expand the filesystem information displayed for selection.

It is important to remember that if you are going to include other users' data or system data, you must authenticate as an admin user when you start ArGest Backup or you will not be able to properly parse and select files in restricted folders.

The right side of the display provides the mechanism for saving backup definitions, selecting archive destinations, selection of Full, Incremental, or Differential backup, and access to the Advanced Backup Features dialog (more on this below). While you may simply select paths or files and then click the Start Backup button, saving your selection as a backup Definition will allow you to recall your current selections at a later date or schedule them for execution at a later time, or on a repeating schedule.

The Advanced View Job options panel also allows you to set up Incremental or Differential jobs in addition to your normal full operations.

**Advanced View
Job Options**

Across the lower portion of the display is the Backup Options group. These options control the automatic verification pass (AutoScan™ Verify), whether to include mounted remote volumes, whether to send (and to whom) email status reports, and whether to include operator-specified metadata for the backup operation.

Advanced View Job Type and Run Options
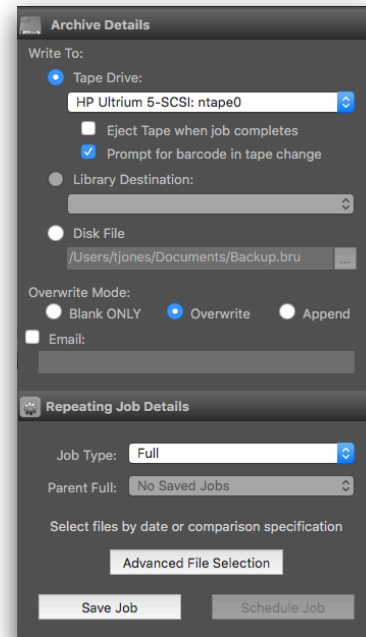
Saving Job Definitions

To save a Backup Definition, select paths or files and add them to the lower list box by dragging the selected items into the lower listbox or by double-clicking each selected item as described above. Click the "Save Definition" button from the right side of the panel.

When the Save BRU Definition dialog appears, You may select an existing definition and overwrite it with the current settings or enter a new name. Additionally, if you will be executing the job via the scheduler, the Incremental and Differential variations on the job can be created automatically if the "Create Default Incremental and Differential Jobs" checkbox is checked.
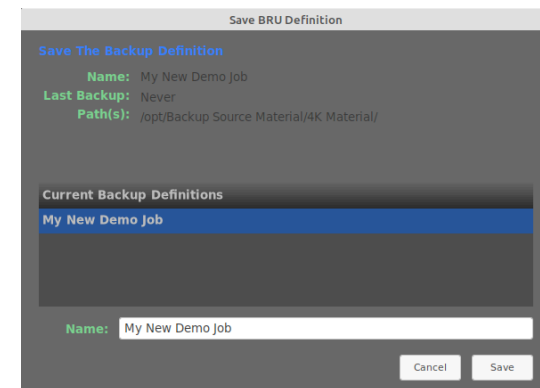
Save Definition Dialog

Once you've saved a backup job, ArGest Backup will ask if you wish to schedule the saved job.
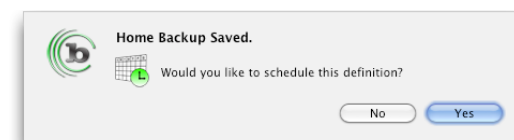
Figure 3 – Schedule A Saved Definition

Clicking "Yes" will take you to the Scheduler panel. We will describe this below.

Once you have saved one or more backup definitions, the Load Definition popup menu can be used to reload the selections saved in the definition.

If you wish to clear your existing selections, you can double-click the entries in the lower list box or click to select the entry that you wish to remove and hit the Delete key. Click the "Clear Selection" button to clear all entries and return the Load Definition menu to the default setting.

Clicking the "Delete Definition" button will allow you to delete a previously saved definition. If you delete a definition, any saved schedules associated with that definition will also be deleted. Note that deleting a backup definition does not affect your ability to restore files backed up using that backup definition.

Depending on the devices available on your system, one, two, or all three of the archive destination selections will be available. "Disk File" will always be available, "Library Destination" will only be available is you have a tape library attached to the system. "Tape Drive" will be available if either a tape drive, or a tape library is available.

To use a disk-based archive file, select the Disk File button and enter the filename that you wish to save to. If you do not provide a path, the file you list will be created in your Documents directory. Also, for ease in later recognizing previous BRU backups, we recommend using an extension of ".bru" – for example: DailyBackup.bru.

If you have a standalone tape drive, select the Tape Drive button.

Library Destinations

For tape libraries, you need to define a Library Destination. If you have a library attached to your system, BRU will recognize this and take you directly to the Library Destination dialog to create your destinations as shown in figure 4. A destination may be all of the tapes in a library, a single tape, or any set of contiguous slots.

*An important note about dealing with the tapes within a destination, you **MUST** treat all tapes in a destination as a single set and not treat the tapes as separate entities. Therefore, **when loading or unloading the tapes in a destination, you must remove and replace all tapes in the assigned slots to change or reuse the destination. You should never remove or replace "some" of the tapes in a destination**.*

If you need to modify the slots assigned to a destination after it has been used, it is also very important that you change the tape(s) in the destination as the slot handling will change and confusion over the media used could result in tape handling issues for the humans involved.

We also do not recommend using perpetual appending of incremental data to a library destination as you will eventually end up needing to replace the media within the destination. The current version of ArGest Backup will allow you to append data to a tapeset in a destination as long as the current operations have not exceeded the capacity of the original media within the destination. Once operations to a destination's tapeset have exceeded the capacity of the tapes that are originally loaded, ArGest Backup will no longer allow you to append to members of that tapeset. If you require this type of operation, our BRU Server solution is designed specifically to support it.

To put it simply, once you define a ArGest Backup library destination, treat it as one tape. If you must change the destination's slot assignments, remove any existing media and replace them with fresh tapes. Do not develop a backup or archival scenario that assumes you will be able to append to a destination's tapes forever.

!! Removing a portion the tapes from a destination's assigned slots will confuse ArGest Backup's tape

handling logic and potentially result in unexpected overwriting of data on tapes within the destination.

Library Destination Warning Dialog

If you have not created a library destination, the warning dialog above will be displayed. Clicking "OK" will display the "***Library Destination Manager***" dialog as shown below.
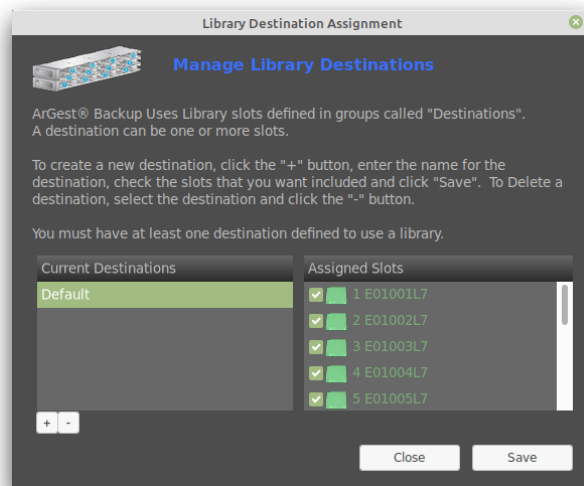
Library Destination Manager

Click the "+" button to create a new destination. Provide a name for the destination and then click the checkboxes next to the slots that you wish to include in the destination. ArGest Backup requires that destinations are contiguous and that slots do not overlap between destination. Click the Save button (or hit Enter) to save the destination.

Your saved destination will now be available in the "Library Destination" popup menu when selecting a Library Destination for backup.

Backup Label and Overwrite Selection

Backup Overwrite Warning

You may continue to overwrite the known backups (their catalog entries will be deleted if you do), change the status to Append – adding the new backup to the existing archives, or you may cancel the backup operation and return to the Backup panel to modify your choices.

Backup Estimation Dialog

If you selected the Estimate Backup option, the Estimate dialog will be displayed. Please note that the values displayed are an estimate and the actual data and file count may be slightly different once the backup is completed.

Once the estimate completes, you may cancel the backup operation, click the Ok button to continue, or allow the 30-second timer to expire, which will automatically start the backup operation.

clicking the

p" button to

tape is

on is a

library, the drive will be checked and if the tape is a member of the selected Library Destination, the backup will proceed. If the drive is empty, or the current tape is not the first tape of the Library Destination selected, the proper tape will be loaded and the backup will proceed.

The label and overwrite dialog will be displayed. Enter a human-readable label in the text entry field. This label may be up to 63 characters long. Also, to estimate the amount of data and number of files processed, select the "Estimate Job" checkbox. You may choose between overwriting and appending the data to the tape. If we were using a disk file for this demonstration, these options are disabled and only overwriting is allowed.

After you click Continue, if you have selected to overwrite the backup archive destination, the catalog data files will be checked against the current tape to determine if you are about to overwrite known backups. If the current media contains a known backup, the overwrite warning screen will be displayed as shown in figure 7.

Backup Progress Dialog

During the backup operation, the Backup Progress dialog will be displayed. The status of the current backup will be shown along with a progress bar that provides a visual clue as to the progress of the backup.

Additional Advanced Options

BRU provides many unique options for file selection when executing a backup operation. These including being able to select files based upon a creation or modification time newer than a specified date or by using the creation time (ctime) of a reference file on your system. Additionally, you may revers this criteria and specify to select file whose mtime or ctime is older than the specified date or references file.

To access these options in ArGest Backup, click the "Advanced Selection" button on the lower right side of the Advanced Backup panel.
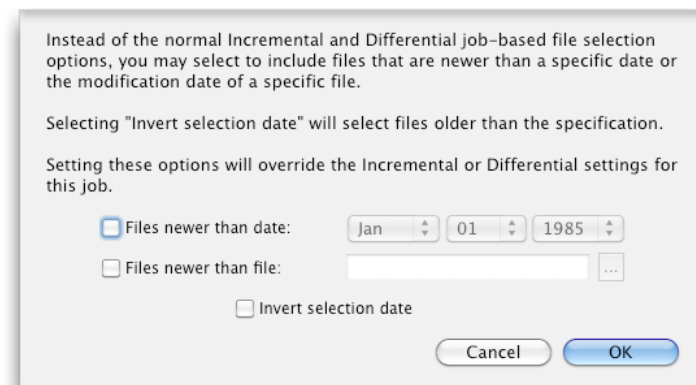
Advanced Selection Dialog

By setting options in this dialog, you can apply out-of-job settings for handling a set of files and folders based upon date and time information that you specify. These types of jobs do not really qualify as Incremental jobs since they may not have a prior reference Full job, but rather are simply defined to contain files created or modified after a specific point in time (or before if you elect to invert the selection criteria).

Once you have set the criteria for selection, you may specify any group of volumes, folders, or files in the main interface and only the content within the selected data that match the criteria set in the "Advanced Selection" dialog will be included in the final backup.
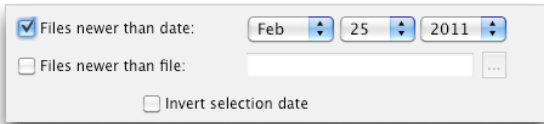
Advanced Selection - Date Example

For example, if you set the options to use "Files newer than date:" with a date of Feb 25, 2011, and then select your entire system drive as the source, the backup operation will process your entire system disk and only select those files that have ctime or mtime values equal to or newer than February 25th. Because we use the filesystem metadata stored in the filesystem on the disk as we traverse the filesystem rather than a separate pre-scan of the filesystem, this operation can occur in a very short period.

Estimate Media

After you have selected the data for archival or backup, clicking the "**Estimate Media**" button will run a BRU estimate of the data selected and provide an estimate of the number of tape volumes required for the job. The values presented are based on the media's native, non-compressed storage capacity. Additionally, the estimated number of tapes will only be displayed if you are using an LTO-4 or LTO-5 drive. In other situations, the estimated total of BRU data to be written will be displayed and you will need to determine the number of tapes based upon your drive and media capacities.

Since it is not possible for BRU to estimate the level of compression that a given data set will achieve on tape, we recommend that you only use the native capacity of your media when determining the number of tapes that will be required.

Media Requirements Estimate Dialog

In the case of a Doubler mode archival operation, you will be provided with the total number of tapes for each job and for the jobs combined.

Complete Final Cut Pro X Environment Support

Starting with ArGest Backup 3.1, there is a new option that will automatically locate all FCP X Projects and Events as well as any referenced media within events to provide a robust and complete backup or archival of your FCP X environment.
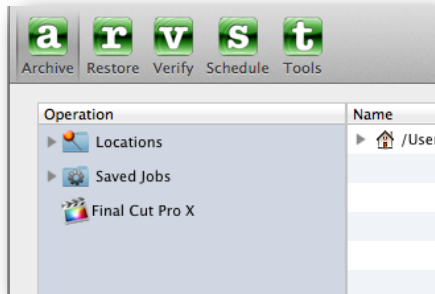
FCP X Backup Option

If you have Final Cut Pro X installed on the same system as ArGest Backup, each time that you start ArGest Backup, the Final Cut Pro X option will be added to the Operation list on the Advanced View window or to the Saved Job Definitions popup menu on the Wizard View window.

Once either of these options is selected, you will be prompted with an options sheet that will allow you to specify what contents will be included in the backup.
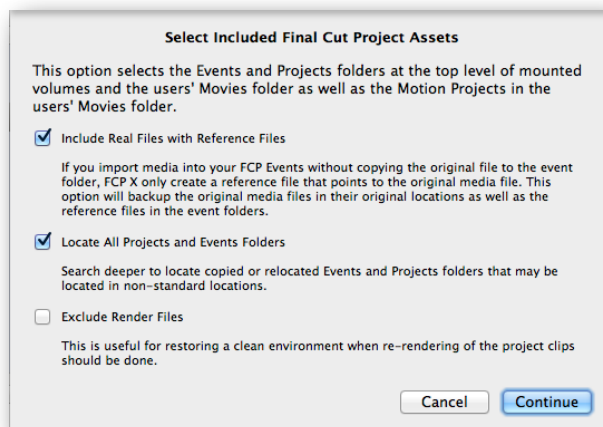
FCP X Content Selection Options

The first option - "Include Real Files with Reference Files" - allows you to include original, hi-def media in the backup if you have chosen to not copy the original files into the Final Cut Events when you imported them. In that situation, FCP X only adds reference files (aliases) within the assigned events while leaving the original files where they are. This is good for keeping your principle media volumes from overfilling, but it is not good if you only backup the actual events as the real files will be missing. Unless you are sure that all of your original media files are being copied into your events when they were imported, it is best to leave this option selected.

The second option - "Locate All Projects and Events Folders" - instructs ArGest Backup to search deeper than the default locations for Projects and Events folders. By default, FCP X places Folders and Events folders at the root level of any mounted volume where you create new events or projects. However, it is also possible for you to copy or move these folders to alternate locations. In that case, BUR PE will scan all mounted volumes for the "Final Cut Events" and "Final Cut Projects" folders in non-standard locations. If you have manually moved or copied these default folders to alternate locations, this option will allow ArGest Backup to locate the folders for inclusion in the backup.

The last option - "Exclude Render Files" - automatically tells ArGest Backup to ignore any rendered output in the Projects folders. This can reduce the amount of data included in the backup operation if you are tight on tape space. Additionally, in an emergency restore situation, you can re-render all of these files. This choice is off by default, but if you are creating an archival set of tapes, the render files may not be a requirement for long term project and environment storage.

After specifying your options. ArGest Backup checks the status of your system's Spotlight indexing. If any of your attached volumes have Spotlight indexing disabled, we will report which volumes are affected and warn that searching these volumes could take much longer than if Spotlight were enabled.

Spotlight Index Warning

While ArGest Backup will still be able to locate the appropriate folders on non-indexed volumes, it may take longer than with Spotlight indexing enabled.

Once your system is scanned, your backup will include all folders that FCP X would normally access as well as the "Motion Templates"; the
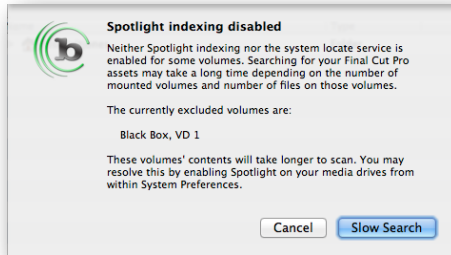
folder that stores any custom transitions, titles, generators and themes that you may have installed.
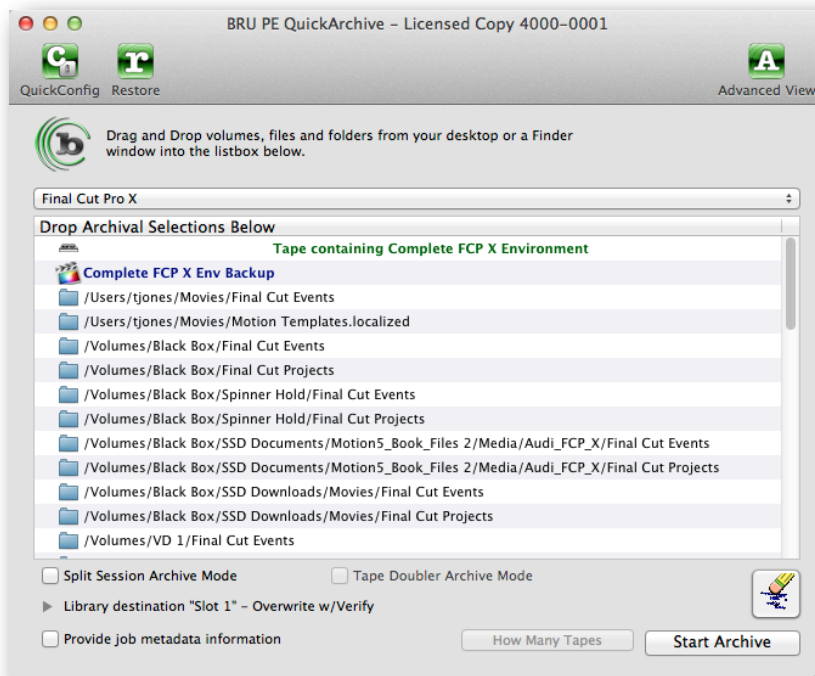
FCP X Complete Session in Wizard View

Including additional files / folders

As with all ArGest Backup jobs, you may add additional files / folders to this job by simply dragging them from your desktop / Finder. While you can do this, for easier archive management and later restore selection, we recommend

<div align="right">

7
Restore Operations

</div>
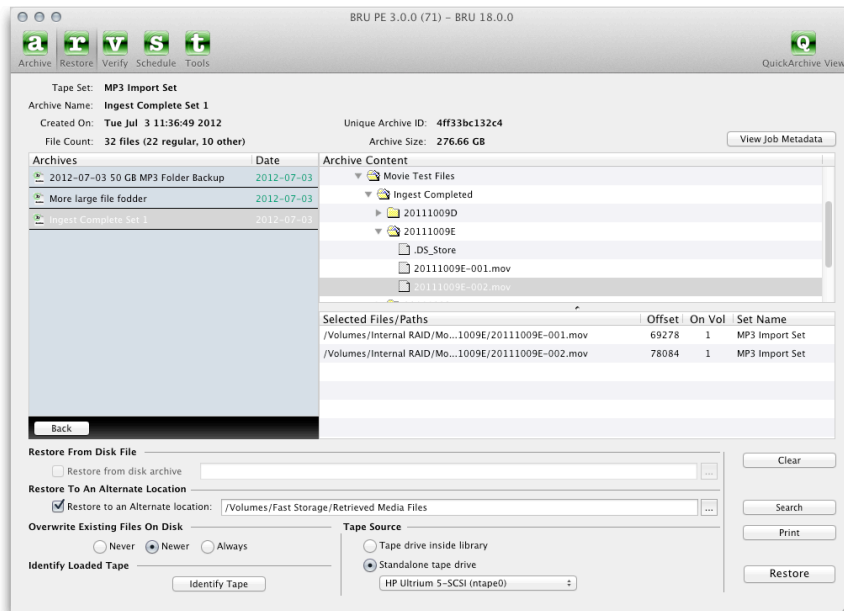
The Restore Panel

The Restore panel provides you the mechanism for selecting the archive and the paths or files within that archive for restoration. You may select a single file, multiple paths or files, or entire archives for restoration. Selection is made by double-clicking the files or folders in the upper list to add them to the lower list.

Normally, when restoring data from an archive, BRU will only overwrite files that exist on the disk if the disk file is older than the file in the archive. However, you may modify this behavior and elect to never overwrite existing file or to always overwrite existing files. Additionally, if you wish to keep the existing disk file and restore the files from an archive, you may specify an alternate restoration path.

Once the selected archive's catalog is loaded into the upper right list box, the quickest way to restore files of interest is to use the search function as shown in the figure below.
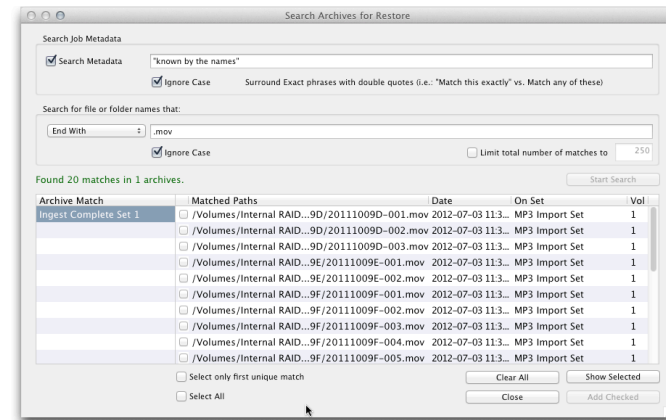
The Restore Search Function

As demonstrated in the QuickStart chapter, this allows you to quickly locate the file or files that you are looking for without manually scanning through what could be hundreds of thousands of files.

If you choose to manually search through the files in the "Archive Content" listbox, locate the files in the upper right list box and double click to add them to the "Selected Files/Paths" list box. Once you have the files or paths selected for restore, click the Restore button to begin the process.

Required Tapes Dialog

BRU will prompt for any required media, or automatically load the appropriate media if you are using a library, and begin the restore. Because BRU tracks where each file is located on a tape, it uses Quick File Access (QFA) to quickly position the tape to the starting location for your requested data. In most cases, this means that your restore will take minutes even if it took many hours to actually complete the backup.
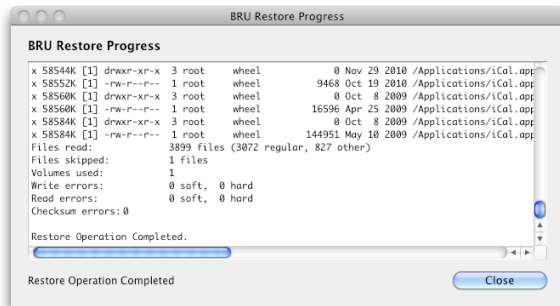
A Completed Restore

After the restore is completed, click the Close button to return to the Restore Panel.

Restoring to an alternate location

By default, BRU will always restore files from an archive to their original location. This is good for restoring files to replace lost or unexpectedly modified files, but it will result in the original file being overwritten by the restored file.

If you need to restore a file without overwriting the current copy on your system - to compare two versions of a color-corrected film clip or compare different versions of a contract, for example - you can elect to restore the file in the archive to a different location.

To change the destination for the restore, check the "Restore to Alternate Location" checkbox in the lower section of the Restore panel. You may then manually type the Unix path to the folder that should be used as the new target destination, drag and drop a folder from the Finder, or click the "..." button to choose the folder from a selection dialog.
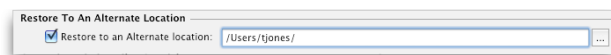
Restore to Alternate Location Field

When the restore operation is run, the files will be restored into the folder specified in the text field.
When the restore completes, there are two resulting paths possible - the original path appended to the alternate restore path or the Original path trimmed so that the top-level path element of the original path in the archive is trimmed to the alternate path specified. This is controlled in

the Preferences -> Restore options setting called "Auto-trim original folder path"
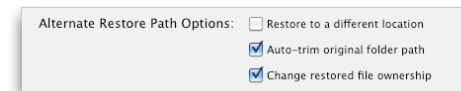
Alternate Restore Path Options

Recover while trimming original path

This mode will trim any path elements from the selected data and restore the chosen files or folders directly into the alternate path. For example, a folder that was originally located at:

/Volumes/Storage 3/Project/Dailies

Being recovered into

/Users/me/Project/

Would result in a restored file layout of

/Users/me/Project/Dailies

Recover using full original path

This mode will retain the full path of the original files or folders and add them to the path selected as the recovery location. Using the previous example, the resulting recovered data would be located at:

/Users/me/Project/Volumes/Storage 3/Project/Dailies

It's important to realize that in this second case, BRU will restore all the original properties and permissions and the resulting recovered files or folders will be hidden in the Finder since the original **/Volumes** folder is hidden in Finder by default.

Examples of paths that exist under the **/Volumes** directory that would be affected by this are external drives such as Xsan Volumes, FireWire, USB, SMB, AFS, NFS, SAS, eSATA, locally attached Fibre Channel RAID devices - basically any volume available to your system that is not the system root volume.

To locate the files after such a recovery has completed, you may either use the Finder's "Go to" command or the Terminal with the **'ls'** (ell ess) command.

To use Finder to get to the newly restored files, press ⌘-SHIFT-G (command-shift-g). When the "Go to..." window appears, type in the full path to the restore point and append **/Volumes** to the path. For example, if you restored your data to **/Users/me/Project/**, then the full path you need to enter is:

> **/Users/me/Project/Volumes/**

Then press [ENTER] and you will see the **Storage 3/Project/Dailies** contents that was previously hidden. You may now move the restored data to a visible folder.

If you get an error, check your path and make sure capital letters are used when needed as well as making sure that the full path to the files was entered correctly.

Restoring from LTFS Volumes

Unlike BRU tapes, LTFS volumes can be accessed simply by copying the files on the tape back to he host system. However, by default LTFS volumes re only known when they are mounted on the system. By using ArGest Backup's LTFS import feature, you can include all of your LTFS volumes in your ArGest Backup Restore environment.
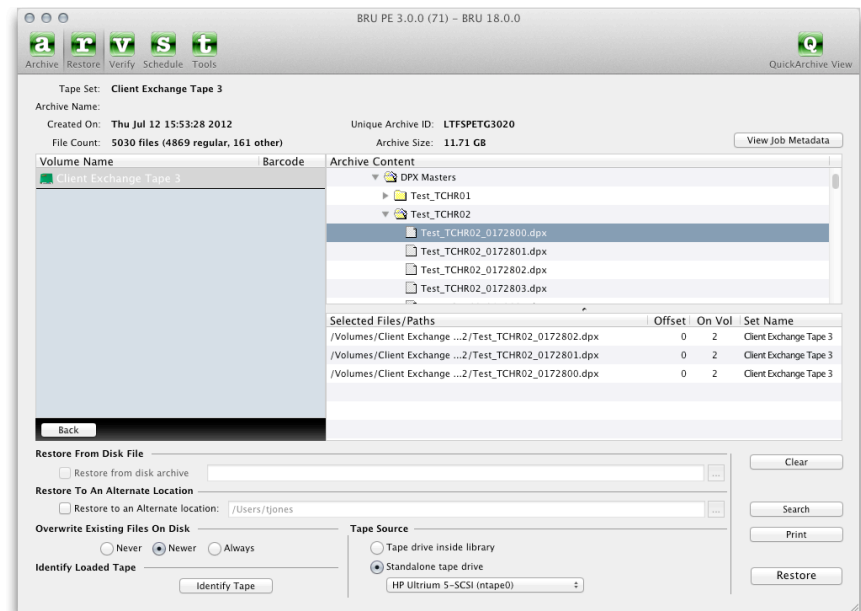
Restore Panel with LTFS Volumes

If you have LTFS volumes imported into your working ArGest Backup environment, they will show up in the Archives list under "LTFS Volumes". Wen you expand the LTFS Volumes section, all of the known LTFS volumes will be displayed according to the name of the tape. All offline volumes will be displayed with a ghosted icon and a mounted volume will be show with a normal icon as shown below.
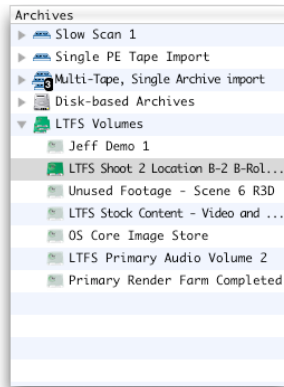
LTFS Volumes In the Archives List

Once you select an LTFS volume and locate the files or folders you are looking for, clicking the Restore button will result in one of two results:

If the LTFS volume required is currently mounted, the folder containing the first entry in the restore selection list will be opened and displayed on your desktop.

If the LTFS volume required is offline, you will be told which tape (by both its label and LTFS volume ID) is required and prompted to mount the volume to restore the files to your system.

Once the appropriate volume is mounted and available, recovery of the files is a simple drag and drop operation.

8
Verify Operations

Verification is one of the most important and probably the most underused function of any backup process. Unless you verify that data you think got onto your backup media actually made it there intact, your backup may not be worth the media it's written on. Between the reading of the data from your filesystem to the actual writing of that data onto your archive media, there are any number of places in the system where the data stream could become corrupted.

With many backup tools, a verify pass compares the archive content with the actual disk files. This means that your backup will take twice as long as is required to perform the backup without the verification – this is the leading reason listed by System Admins for not running a verify pass. BRU changes this. Because BRU uses an on-tape 32 bit CRC, your verify pass can take place totally out-of-band. This means that a 4 hour backup only needs to access your filesystems for the 4 hours of the backup – the verify takes place solely between BRU and your archive media.

Additionally, this mechanism even allows you to validate an archive set weeks or even years after it was created – even on another system or a different operating system.

While the default for any BRU backup is to automatically verify the archive once the backup completes (AutoScan Verify), BRU also allows you to manually verify an archive.
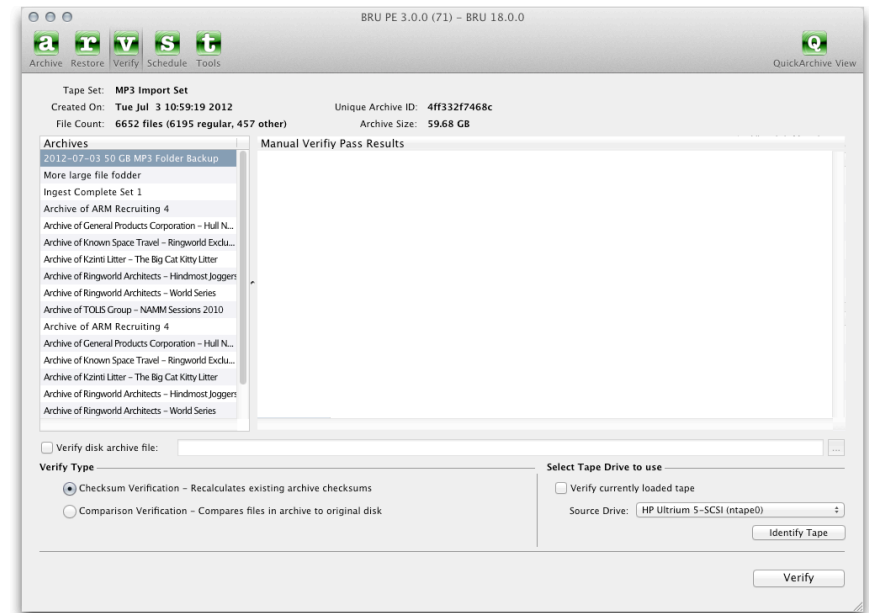
The Verify Panel

BRU will display all known archives in the left side list box. Select the archive set that you wish to verify and the choose either the Checksum Verification button (the default), or the "Comparison Verification" button and click the "Verify Archive" button.

If you are using a standalone tape drive, BRU will prompt you to insert the first tape. If you are using a library, BRU will load the tapes from the assigned destinations slots and proceed automatically.

The right side list box will display any discrepancies that are found during the actual verify pass.

Since we promote CRC, or Checksum verification as the best verification method, why do we offer a Comparison option? Using a disk-to-archive compare is a good way of determining if files that shouldn't have changed on a system have been changed and, therefore, potentially compromised. We have many users that create baseline backups of their systems' configuration files and if foul play is suspected, they perform a comparison verification pass to check the system status.

If you wish to verify an archive that is unknown to this installation of BRU, select the "Verify Currently loaded tape" checkbox after selecting the

active tape device. BRU will verify the contents of the tape currently in the drive as well as prompt for additional volumes if it involves a multi-volume archive, but this tape will remain unknown to ArGest Backup unless it is imported using the separate Import Tool.

<u>9</u>
<u>Scheduling</u>

Scheduling functionality and documentation will be coming in a future
ArGest Backup update.

<div align="right">

<u>__10</u>
<u>The Tools Tab</u>

</div>

The Log Panel

The Tools panel is divided into 4 panels and provides a view of your environment as well as options for viewing your execution log, manually controlling a library and tape drive, retrieving system information, exporting or importing your full catalog and tape info, backing up or restoring your full environment, checking for application updates, and determining system information.

The Environment Panel
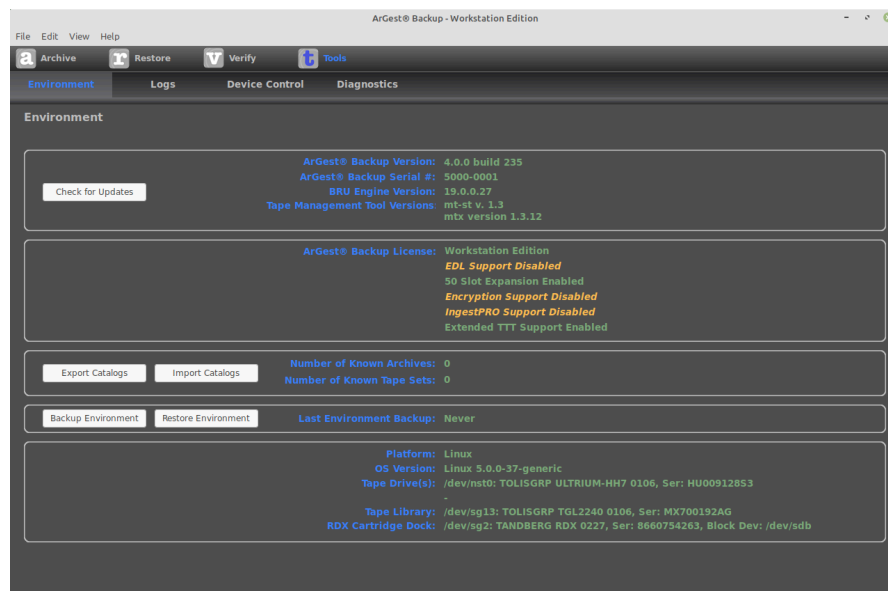
Tools - Environment Panel

The Environment panel is like a simple dashboard, displaying your application information and licensed elements, tape set and archive counts, top level system information, as well as providing import and export of your catalog and tape information, backup and restore of your full ArGest Backup environment, and checking for product updates.

When contacting support, we will ask for the information on this page.

The Device Manager Panel
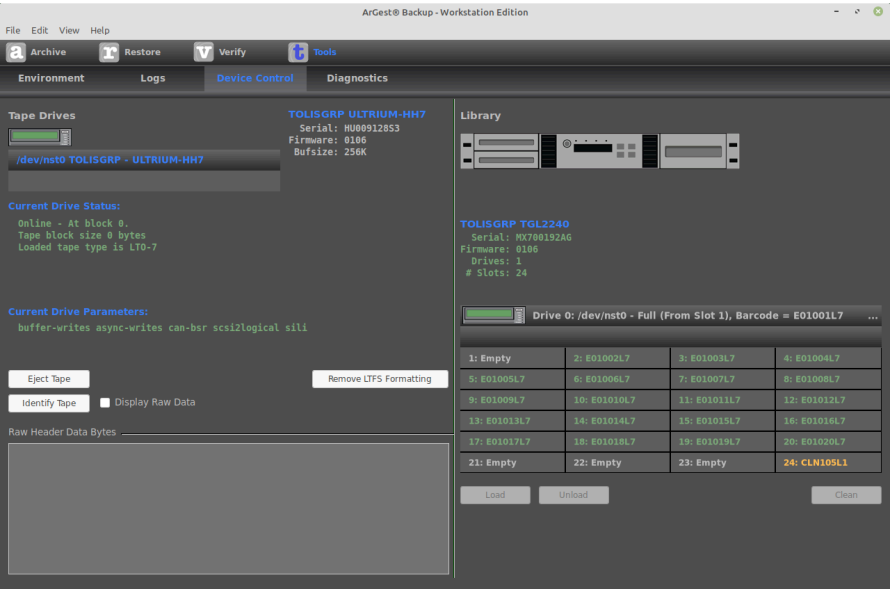
Tools - Device Manager Panel

The Device Manager panel allows you to check the status of your tape or library devices, remotely eject tapes, identify the contents of an inserted tape - including viewing the raw data in the first 4K of a loaded tape, and a shortcut to removing LTFS formatting from a tape to allow it to be used for normal data backup operations.

We discuss the two separate sections below.

Controlling a Tape Drive
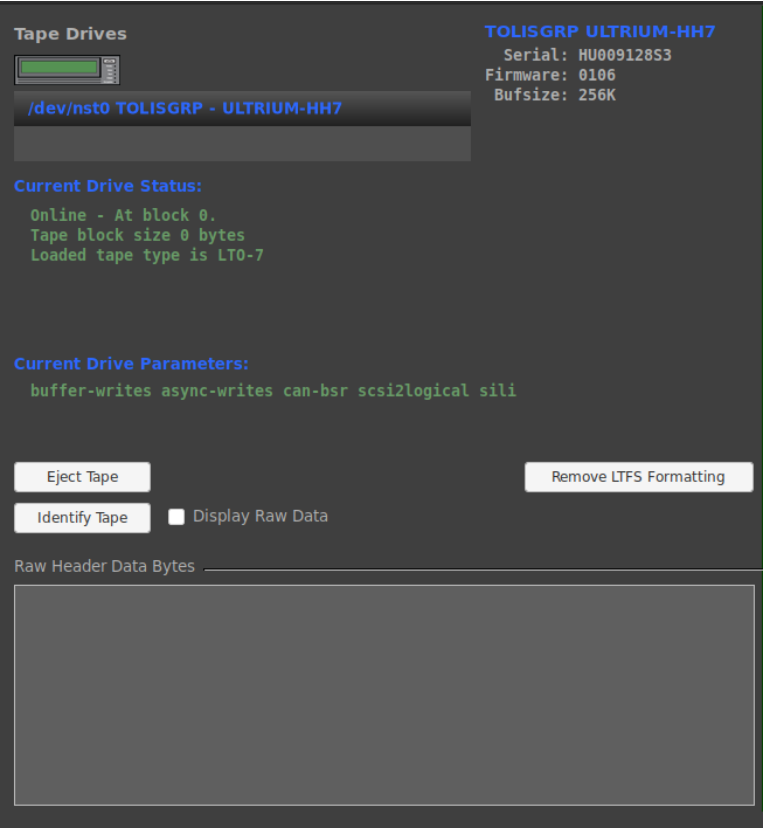
Tape Drive Control Section

For a standalone drive, this panel provides general device information, tape load-state and media type, specialized drive parameter settings, allows you to remove LTFS formatting to return an LTFS tape to normal data format, remotely eject a tape, and identify a loaded tape with the option of displaying the raw data from the first 4K of the tape.

Controlling a Library
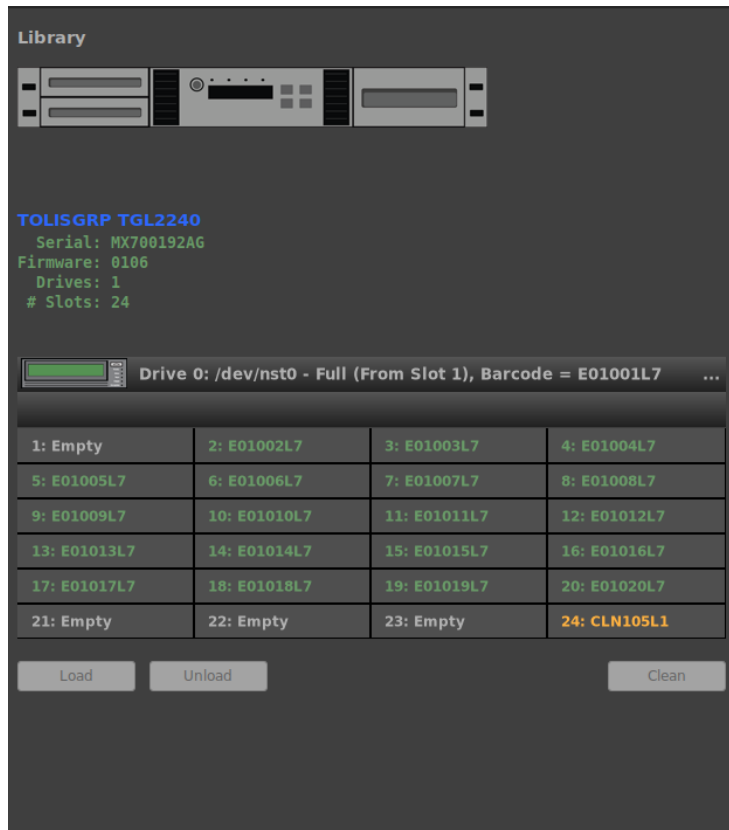
Library Control Section

The Library options panel displays the current load status of your library. You may use the controls to manually load or unload the tape device and import or export tapes to or from the library (if supported by your library).

To load a tape from a slot into a drive, select the Drive to load and then select the slot to load the tape from and click the "Load" button. The "Load" button will only be enabled if the drive that you have selected is empty and the tape slot you have selected has a tape loaded.

To unload a tape from the drive to a slot is just the opposite. Select the loaded drive and an empty slot and click the "Unload" button.

The Diagnostics Panel
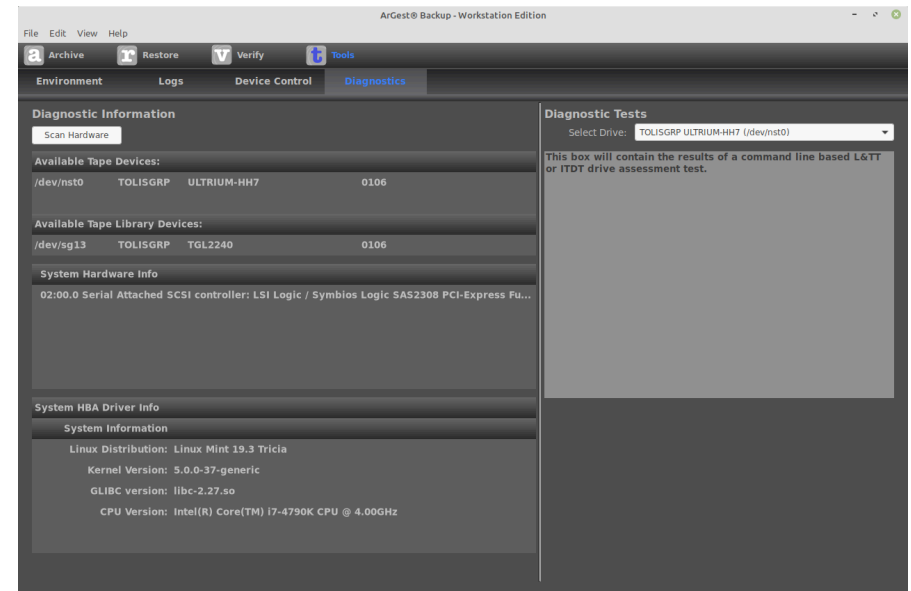
Tools - Diagnostic Panel Linux
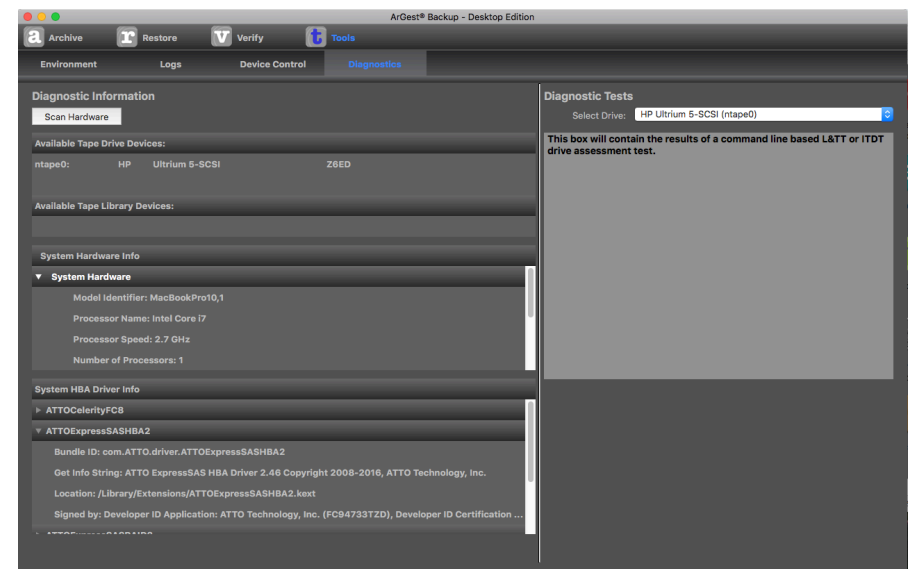
Tools - Diagnostic Panel macOS

Sharing ArGest Backup Tapes

ArGest Backup offers a very easy to use method for sharing tapes with clients or other ArGest Backup installations. To simply track tapes between ArGest Backup installations, you can export your archive information from one system and copy it to another system. Doing this allows all systems involved to know the contents of the tape(s) in question.

To provide a solution for clients that may not be using ArGest Backup in the daily operations, ArGest Backup offers the ability to create what we refer to as the BRUsetta Stone CD. Like history's Rosetta Stone, a BRUsetta Stone CD will provide the recipient with all of the information required to recover the data on the associated ArGest Backup tape(s) – including an installable copy of ArGest Backup for OS X.

Exporting Archive Catalogs

There are two ways to export your archive catalogs -

When a backup job completes, A dialog is displayed with the results of the job and instructions for naming your tape(s). You can select to export the catalog information (creating a zip file that can be copied to another ArGest Backup system), print the catalog information (literally to a printer or to a PDF file), and create a BRUsetta Stone CD.
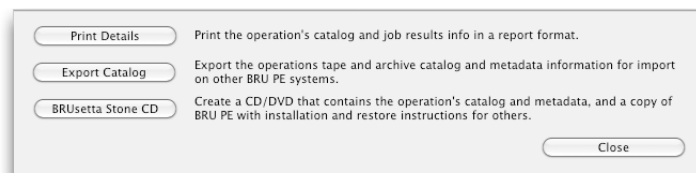
Catalog Output Options

Alternatively, you can select a tape set on the Restore panel and right-click (CTRL Click if using a one-button mouse), and select either "Export Catalogs" or "Create BRUsetta Stone CD" from the popup menu.

This menu provides the same options

presented at the end of a normal archive operation above.

If you elect to simply export the catalogs, you will be prompted for a destination folder and a zip file containing the tape and archive information about the current or selected job will be created.

If you elect to create a BRUsetta Stone CD, you will be asked to locate a copy of the ArGest Backup Installer DMG and to create a folder to use to create the actual CD contents.
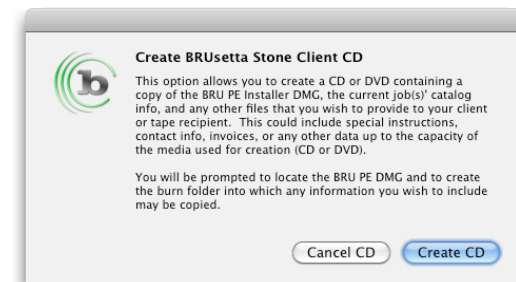
BRUsetta Stone CD Dialog

Once you create the folder, the default files will be added to the folder:

- ArGest Backup Installer DMG
- The Tape and Catalog info zip file for the tape and archive(s)
- A ReadMe file that describes the contents of the tape and provides basic guidance to recovering the data from the associated tape(s).

You may add any additional files that you require - Client-specific instructions, your contact info, an invoice for the work, additional tools the customer may need, and anything else you wish up to the capacity of either a CD or DVD disk.

Additional File Dialog

When you've completed adding your additional data to the folder, click the Continue button to burn the folder contents to the CD or DVD. The disk created is in ISO9660 format and will be compatible with any operating system.
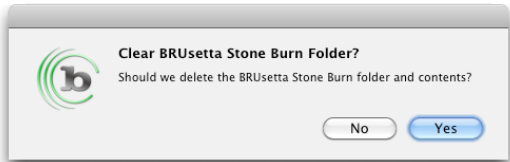
Clear BRUsetta Stone Folder

After the disk burn completes, you may elect to clear the folder or retain it for additional burns using Disk Utility.

Printing Catalog Info

If you select to Print Catalog from either option, the Print Configuration dialog is displayed to allow you to select what parts of the catalog info is included in the printed report.
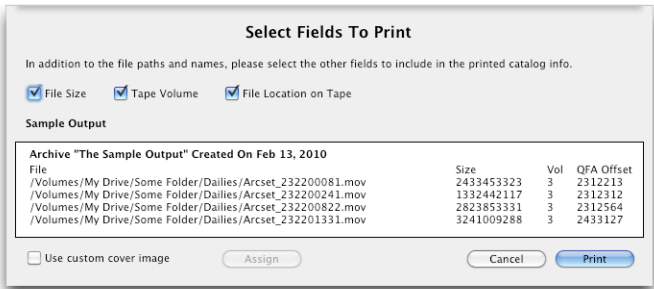
Catalog Print Configuration Dialog

The file path and name will always be printed, but you may choose whether to include the file size, tape volume, and file block offset along with the file information.

By checking the "Use custom cover image" option, you can choose a custom picture for the cover of the report. This image can be any size and type - GIF, JPG, TIFF, or PNG - and it will be scaled to fit in the area of the cover page (the image of NGC2170 in the figure below was originally 2004x1336) .
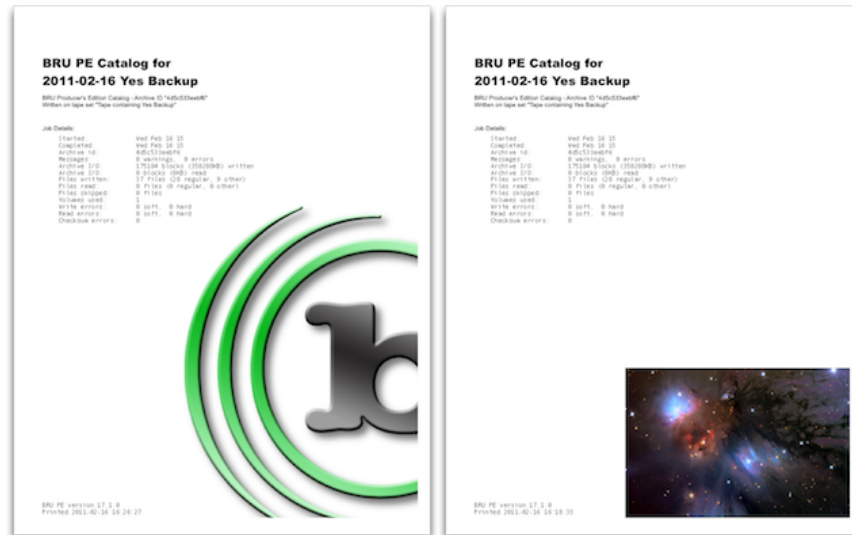
Example Cover With Default and Custom Image

You may use any settings in your normal printing environment including saving as PDF or Postscript as well as physically printing.

Recommended Workflow

If you are working with a data set that will be provided to a customer or stored offsite for an extended period, we recommend the following workflow.

- Complete your archival operation
- Select the "Print Details" option and create a PDF of the catalog
- Select the "Create BRUsetta Stone CD" option and add the PDF created to the folder
- Burn the CD and store it with the tape(s)

This workflow will assure any recipient of the tape(s) will have everything that they need to recover the data in the archive(s) - short of a physical tape drive.

<div style="text-align: right">

12
</div>

Advanced Command Line Operation

Overview

BRU for OS X's scriptable command line operation may be used on any system. This chapter describes tips on getting started using BRU's command line quickly.

Tape under OS X

Unlike other UNIX environments, OS X does not offer the standard /dev entries for tape drives. Therefore, we manage all tape I/O within the BRU application and its companion utility 'tapectl'. To simplify tape discovery, the tapectl utility offers a 'display' option, which will discover and enumerate the available, compatible tape drives attached to your system. BRU and the provided tapectl utility will support tape drives connected to SAS, Fibre Channel, SCSI, Firewire, iSCSI, or USB interfaces with no special considerations on your part. No matter what the interface type, the first tape drive located will be named tape0/ ntape0, the second - tape1/ntape1, and so on. We have tested the environment with 5 tape drives attached to a single system simultaneously, including 1 USB, 1 Firewire, and 3 SCSI drives. To discover the drives recognized by your system, use the tapectl command's display mode:

An example of output within the Terminal window would be:

```
# tapectl display[ENTER]
Available Tape Devices:
  ntape0: HP Ultrium 5-SCSI Z39W
```

For the **tapectl** command, we always access the non-rewinding version of each drive. Therefore, to get the status of the 2nd drive attached to your system, you would use the **ntape1** device as an argument to **tapectl**:

```
Engineering-MP:~ tjones$ tapectl -v status
Vendor = HP        , Model = Ultrium 5-SCSI
Revision Level = Z39W
BlockSize: 0
At block 0
Currently at BOT
```

In this case, we are querying the HP LTO-5 (ntape0—the default device) attached to the SAS bus and there is a tape inserted positioned at block 0, or the beginning of tape (BOT).

For most tape drives, you may use the default settings applied by both BRU and tapectl. However, if you have more than one tape drive attached to your system, the order of discovery will indicate the numeric value assigned. For instance, if I have a USB tape device and a SCSI tape device, the USB device will be discovered first, the Firewire device next, and any SCSI, SAS, or F-C devices last. This places it in the position we refer to as tape0. The SCSI tape device would then be attached as tape1. As you can see in the '**tapectl display**' example above, there were 3 devices attached to our G4 - a USB OnStream ADR2-60, an HP DDS3 SureStore 24 DAT on a Firewire to SCSI adapter, and a Quantum/Benchmark VS80 DLT on the SCSI bus. Additionally, on the SCSI bus, drives with higher SCSI ID's will be placed in our device list first. If we have two drives set at ID's 5 and 4, the drive at ID 5 would be the lower numbered tape device.

The full array of tape management functions available with the provided tapectl version can be determined by typing 'tapectl' with no arguments:

```
# tapectl[RETURN]
tapectl [-V] [-v] [-f dev] arg [count]
Version 8.0
  Devices are known by the names ntapeX where X is the
device order number
  starting with 0 and incrementing for each tape device
found.
  Options are:
  display  - displays available tape devices
  status   - display current drive status
  header   - read the first 4K of a tape and display on
               stdout
  fsf x    - forward space filemark
               (x = count; 1 is default)
  bsf x    - back space filemark
               (like fsf, but backwards)
  seod     - space to end of data for appends
  erase    - short erase a tape
  lerase   - long (secure) erase - takes hours!
  rewind   - rewinds a tape
  rewoff   - rewind the tape and unload it
  reten    - retention a tape (QIC and Travan tapes)
  tell     - report current logical tape block
  seek x   - seek to logical tape block 'x'
  setblk x - set drive block size to x
               (x is required, 0 = variable block)
  alert    - display TapeAlert information if available
  unload   - same as rewoff
  -f dev   - device to access - (ntape0 is default)
  -v       - additional verbosity while working
  -V       - version and copyright information
```

Further Descriptions

```
display: Displays the recognized devices attached to
         your system
status:  Reports the status of the selected device
         (ntape0 is the default).  Output looks like:
              Vendor = OnStream, Model = ADR Series
              Revision Level = 6.04
              Medium Type: 0x0 (empty)
              Density Code: 0x0 - default
              BlockSize: 32768
              Drive Not Ready, No tape inserted
fsf [x]: Moves the tape forward by x number of filemarks
         - default is 1. When the motion completes, the
         tape is positioned on the end of tape side of
         the selected filemark ready to read the
         selected backup set or write new data (if you
         are at the end of data mark).
```

```
bsf [x]: Moves the tape backward by x number of
         filemarks - default is
         1.  When the motion completes, the tape is
         positioned on the end of tape side of the
         selected filemark ready to read the selected
         backup set.
seod:    Moves the tape to the end of currently recorded
         data.  Use the option to position that tape for
         appending additional backups onto a previously
         written tape.
erase:   Short erase blanks the tape in preparation for
         new data.  Even though his only writes the end
         of data mark and null data zone onto the tape,
         any data beyond that point will be
         unrecoverable.
lerase:  Provides a secure erase of the entire media
         (takes a long time for most drives).  Like a
         short erase, but every tape block is
         overwritten.
rewind:  Rewinds the tape to its beginning.
rewoff:  Rewinds the tape to its beginning and ejects it
         (on soft eject capable drives).
reten:   Fast winds a tape from one end to the other.
         Especially important for QIC, SLR, and TRAVAN
         drives.
tell:    Reports the logical block address of the tapes
         current position.
seek x:  Moves the tape to logical block address x.  The
         value of x must be decimal and is required.
setblk x:Sets the blocksize for the loaded media.  If x
         is zero, this sets the drive into variable
         block mode.  Note that not all drives support
         changing the blocksize.  If the requested
         change is not allowed, and error will be
         reported.
alert:   Displays TapeAlert messages from devices that
         support the TapeAlert message standard.
unload:  Synonym for rewoff Rewinds the tape to its
         beginning and ejects it (on soft eject capable
         drives).
```

Command Line Examples

BRU LE for Mac OS X's command line is designed for maximum functionality from the OS X Terminal command line (or a standard XTerm if you're running X11 for OS X). The following samples are designed to provide an idea of what BRU is capable of. In these examples, please remember to issue a

'`tapectl -f ntape0 rewind`' command to return the tape to its starting point.

Archive creation uses the `-c` option (for create). To backup the full system, simply list / (slash, or root) as the file list argument:

```
bru -cvf ntape0 /
```

This command will backup everything on your system (including any mounted volumes) to the non-rewinding device ntape0. The output will provide a simple listing of the files as they are processed:

```
/bin/pwd
/bin/rcp
/bin/rm
/bin/rmdir
/bin/sh
/bin/sleep
/bin/stty
/bin/sync
/bin/tcsh
/bin/test
/bin/writecnt
```

If you wish to increase the amount of information provided, simply increase the number of −v's (up to 5):

```
bru -cvvvf ntape0 /
The results now look like:
c 2554K [1] /bin/pwd
c 2572K [1] /bin/rcp
c 2602K [1] /bin/rm
c 2626K [1] /bin/rmdir
c 2644K [1] /bin/sh
c 3250K [1] /bin/sleep
c 3262K [1] /bin/stty
c 3292K [1] /bin/sync
c 3306K [1] /bin/tcsh
c 3660K [1] /bin/test
c 3678K [1] /bin/writecnt symbolic link to /bin/bru
```

Since BRU sends all normal output to stdout, you may use standard UNIX-style redirection to save the output of any BRU operation to a file (or pipe it to other commands).

```
bru -cvvvvvf ntape0 / > /tmp/osxfull.log
```

This will create a file in your /`tmp` directory that contains the output results of the BRU operation. Additionally, you may send BRU's operation to the background by adding the standard '&' character to the end of the command line.

To examine the contents of a BRU archive, use the table of contents command `-t`. By default, `-t` automatically prints the names of the files in the archive. However, you may increase this output by increasing the number of v's as above in the create process.

To perform a restore of data, we use the −x option (extract). Assuming we'd completed a backup of our entire system, the following line would restore the /`System/Library/Frameworks`, not overwriting any files that are already on the system (`-E`):

```
bru -xvvvf ntape0 -E /System/Library/Frameworks
```

If we prefer to overwrite the existing files, change the −E to −ua:

```
bru -xvvvf ntape0 -ua /System/Library/Frameworks
```

In this example, BRU will make use of a SmartRestore patterns file to ensure that any 'in use' Frameworks are not clobbered by the restore. In the case a Framework is in use, BRU will relocate the currently busy content and replace it with the restored copy. The system will continue to access the currently open files until either that Framework is no longer needed, or until the next system reboot. For more information, please refer to Chapter 18.

The key to BRU's ability to monitor the quality of a backup lies in its verification capabilities. During a backup, BRU will recognize most common tape errors and report them. This is great for the current status, but what about a tape that's 4 months old? By using BRU's `-i` option (inspect), you can examine the contents on any BRU tape, even if it was not made on your system. This

allows you to ensure the validity of your backups at any time, on any system with a similar tape device and a copy of BRU.

```
bru -ivvvf ntape0
```

This command will list any discrepancies in the archive and ensure that the maximum amount of data is always available, even in the event of hard media errors.

If you are verifying a backup on the same system on which it was created, the differences mode (-d) will let you know of any differences that exist between the system and the backup contents. This is a great method for comparing a potentially compromised system to a known good backup. BRU will report any changes, additions, or deletions on the system.

```
bru -ddddf ntape0
```

Getting Started

The many variations on the options available from the command line can be daunting for a first time user. For most operations, only a small number of options are used.

If you have ever used the UNIX command 'tar', you're already a good way towards understanding the way BRU operates from the command line. To simplify things, let's examine some of the basic properties of any BRU backup:

- What operation are you invoking? Backup, Restore, Table of Contents, Verify or Estimate?
- What backup device are you using? A tape drive, a disk file?
- What file system or selection of files are you backing up or restoring?

Once you know these items, you need to examine the various modes and options that BRU provides. These include:

`-c` – Create (backup) a BRU backup volume

`-x` – Extract (restore) files from a BRU backup volume

`-t` – Table of Contents (file list) of a BRU backup volume

`-i` – Inspect (verify) the contents of a BRU backup volume

`-e` – Estimate the number of volumes a BRU backup will require

`-d` – Difference (compare) the contents of a BRU backup volume with the original files on the file system

`-v` – Verbosity level (up to five v's may be specified)

`-f` – What backup device should be accessed

BRU's command line looks like this:

`bru -mode -[options] [-f device] [path]`

With these basic modes and options, all basic backup and restore functions can be performed. For example, to backup the contents of the entire system to the first SCSI tape drive under Linux, we would issue the following BRU command:

`bru -cvf tape0 /`

This will backup the entire system to the tape drive `/dev/st0`. To examine the contents of a tape made in this manner, we could issue the command:

`bru -tvf tape0`

Notice that we don't have to specify a file path for a listing of the tape.

To restore the entire backup to its original path, we would use:

`bru -xvf tape0`

This call would automatically restore all files to their original locations.

To verify the contents of a BRU backup volume, there are two options available. The first is TOLIS' recommended option as it requires only the tape drive and BRU to operate. This is the inspect (–i) mode.

`bru -ivf tape0`

This verification rereads each buffer block written on the backup volume and recalculates the block's checksum. BRU then compares this calculated checksum with the checksum that was written in the buffer block header. If an incorrect value is detected, BRU will issue a warning that the checksum is bad for the particular file that the error occurred within. However, BRU will restore the available data in the file and any additional data on the backup - unlike other applications which abort at the first sign of a tape error.

The second option is a more widely used method that requires both the backup volume and the original data. This mechanism is called the difference (–d) mode.

`bru -dvf tape0`

This option reads the data from the tape and performs a bit-by-bit comparison with the original data from the file system. Of course, this mechanism will report problems if files have

changed on the file system since the backup was made. This is what makes the inspect mode verification the preferred option for verifying tapes.

Advanced Operations

Now that we understand the basic BRU command line operations, we can examine some of the more advanced options that can make the difference between a plain backup and a backup that provides enhanced functionality unavailable in other backup utilities, including:

- `-L` - Place a human-readable text label on the backup volume
- `-g` - Read and display ONLY the backup volume information
- `-n` - Select files based on a date and time specification
- `-B` - Run BRU in the background
- `-PA` - Change Absolute paths to relative (strip the leading /)
- `-ua` - Unconditionally overwrite ALL files during restore

These options allow you to more selectively control the backup or restore operation performed.

To add the description "Full System Backup" to the backup example from above, we would use:

```
bru -cvVR -L "Full System Backup" -f tape0 /
```

Using bru –g, returns the following:

```
# bru -gf tape0
label:          Full System Backup by Tim
created:        Mon Apr 27 08:55:02 1998
artime:         893692502l
archive_id:         3544aa561537
volume:         1
writes:         3
release:        17.0
variant:        0.3
bufsize:        32768
msize:          0
msize_blks:     0
```

```
serial_number: xxxx-nnnn-n
device:         tape0
user:           root
group:          root
system:         Linux giza 2.0.33 #1 Wed J i686
bru:            Fifth OEM Release
command_line: bru -cvVR -L "Full System Backup" -f
tape0 /
```

As you can see, there's a lot of information in the volume header. The important items here are the label, creation date, volume, and command line. If you wish to use BRU to perform Incremental or Differential backups, you could use 'find' and pipe the results to the BRU command, however, BRU provides a better method for locating files based on their date-time stamps - the '–n' option. Using the –n option, you can pass a standard date string, like "01–May–1999", or you can create a reference file and pass the name of that file to BRU, like `/etc/LASTFULL`.

Here's a sample script (change the device for your system) that performs full backups on Saturday morning and differential backups on all other days:

```
#!/bin/sh
#
# This script performs full system backups on Saturday of
# each week and then performs differential (data changed
# since last full backup) backups on all other days.
#
# It would be run by cron at some time in the evening
# when everyone is logged out of the system for best
# coverage.
####################################################

DOW=`date +%w`
if [ $DOW = 6 ]
then
  touch /etc/LASTFULL
  bru -cvf tape0 -L"Full Backup `date`" /
else
  bru -cvf tape0 -L"Differential Backup `date`" -n /etc/
LASTFULL /
fi
```

For more information on the uses of BRU's command line operations, be sure to review the remaining chapters of this manual and visit

http://www.owc.com.

13
The brutab *File*

Overview

BRU works most effectively when it is properly matched with the backup device it will be using. The proper device settings allow BRU to run at maximum speed and increase its ability to detect and recover from errors.

The device parameters are described in a file that is read by BRU when it begins execution. Normally, the BRU device file is named brutab and is located in the /etc directory. The /etc/brutab file may contain entries for several different devices. This file describes: The name and characteristics of the available archive devices (a description can cover more than one such device). When you provide entries in brutab describing, for example, the size of the archive tape, then the size parameter need not be provided to BRU on the command line when the program is called.

Values (either symbolic or numeric) that your own system generates for errors of various kinds. This allows BRU to respond properly when certain conditions (like end-of-tape or write protect) are encountered. It also lets BRU generate the correct message when an error is detected.

You may have several versions of brutab if you wish. Normally BRU looks for a device parameter file named /etc/brutab. However, you can override this default by setting the BRUTAB environment variable to point to the version of brutab you want to use.

Here's an example of how to set the BRUTAB environment variable (under the Bourne shell):

```
$ BRUTAB=/tmp/brutab.new
$ export BRUTAB
```

The brutab file usually has sample device configurations for many systems on which BRU has already been installed. If your specific device or system is not listed, you can probably find one that is similar. Simply copy the applicable entry and edit it for your device. **NOTE**: The brutab file must contain at least two entries: One describes the capabilities of the default archive device, and a second describes the capabilities of the

standard input device and its error messages. See Appendix A for example brutab file.

The Syntax of brutab Entries

Device entries in brutab consist of multiple fields separated by white space (spaces or tabs) or a new line character. Entries may span more than one line if you enter your shell's escape character (usually a \) immediately before the newline character at the end of the line.

Each brutab entry begins with the device name, followed by one or more parameter fields. All tabs and blanks between fields are ignored. Any line that begins with a # character is ignored and will be treated as a comment (except for global brutab parameter lines, which are described later).

The device name has the following form:

```
device name | ... | device name
```

For example:

```
ntape0
```
or
```
ntape0|ntape1|ntape2…
```

If the device name field specifies multiple devices, each device must have exactly the same characteristics and capabilities. Note the lack of spaces in the device name field.

There are two types of parameter fields used to define the characteristics of a backup device. Fields are either Alphanumeric or Boolean (true/false). An alphanumeric field consists of the parameter name, followed immediately by an equal sign '=' and the value. There must not be any spaces before or after the equal sign. The value following the equal sign may be a number, a symbolic value, or a string. String values must be enclosed in double quotes.

A Boolean field consists of the parameter name only. If the field is present, the corresponding device capability will be enabled (turned ON). If the Boolean field is absent, the corresponding device capability will be disabled (turned OFF).

Here is an example of various parameters and values:

```
size=640K noautoscan pwerr=EIO \
fmtcmd="format /dev/rfd1135ds9"
```

NOTE: That there is no space between the parameter name and the value. Numeric values may be given in absolute form or appended with a scale factor of:

> b or B Blocks (2048 bytes)
> k or K Kilobytes (1024 bytes)
> m or M Megabytes (1024*1024 bytes)
> g or G Gigabytes (1024*1024*1024 bytes)

The letter T may also be used as a modifier after the scale factors M or G. This is commonly used when specifying tape size.

When T is specified, a multiplier of 1000 will be used (instead of 1024). This is useful when using the tape size given by the tape manufacturer. Tape megabytes are measured differently than memory megabytes. A tape that is specified as 150 tape megabytes will actually hold only 143 memory megabytes.

Listed below are the scale factors when the T (for Tape) modifier is used:

> kt or KT Kilobytes (1000 bytes)
> mt or MT Megabytes (1000*1000 bytes)
> gt or GT Gigabytes (1000*1000*1000 bytes)

This is an example of how to specify the media size for a tape with a manufacturer-specified size of 525 megabytes:

```
size=525MT
```

Certain parameter fields (like pwerr=EIO) refer to error codes returned by the UNIX systems. The values for these parameters may be given as digits or in symbolic form (like EIO or ENOSPC).

NOTE FOR PROGRAMMERS: Symbolic error codes are usually defined in the development system header include file /usr/include/errno.h.

brutab Default Values

If you do not know the value of a parameter for your system, simply leave it out and BRU will use a built-in default value. If you're not sure, try using BRU with the default values first. On most UNIX systems, the default

values will work without any problems. If you have trouble with the default values, try using the values from another brutab device entry that is similar to yours. If you cannot determine the proper values, you may want to contact TOLIS Group support for assistance.

The Default Device

If no device is specified on the command line, BRU will use the first device listed in the brutab file. To use a different device, you must either specify the device name with the –f option or set the BRUDEV environment variable to a device name.

The following command line specifies the device tape0:

```
$ bru -c -f tape0
```

If the BRUDEV environment variable is set to a device name, BRU will use that device as the default instead of the first entry in the brutab file. The command shown below sets BRUDEV to ntape1 and will cause BRU to use ntape1 as the default device:

```
$ BRUDEV=ntape1
$ export BRUDEV
```

brutab Fields

The following sections contain names and descriptions of alphanumeric and boolean brutab fields and of global brutab parameters. This is information that may be needed during installation. See both Chapter 2, "Installation" and Appendix E, "A Sample brutab File."

Alphanumeric brutab Fields

These are the alphanumeric fields currently recognized by BRU. The two most important fields are listed first:

size=

Media size in bytes, if known. Zero, if unknown, or if variable-sized media. If a size is given, it must not be larger than the actual media size. For instance, do not specify size=30G if you are using a 15 gigabyte tape.

This option should be used with fixed size devices such as tape drives, removable devices (Iomega Zip or Jaz), and floppy disks. With modern tape drives that provide hardware compression, it is not possible to properly define the amount of data that a cartridge will store - in these cases, please use a setting of size=0.

If you do specify a size other than 0, the specified size must not be larger than the actual tape size. If you set it larger, it may cause confusing error messages. This is because BRU is unable to determine the difference between an end-of-tape condition and a real error. To avoid problems, it is better to set the size too small than too large.

Media size can also be specified on the command line with the –s option. This command line specification will override any value specified in brutab. **IMPORTANT**: The media size must be specified as a whole number, no decimals are allowed. Specifying size=2.2GT is **ILLEGAL** and will not work. Instead, you should use size=2200MT.

bufsize=

Default I/O buffer size for this device. If omitted, the default is 20 kilobytes. If you are using double buffering (–D) this bufsize value should not be larger than the value of shmmax, or the system

imposed limit on the size of a shared memory segment, whichever is smaller. The buffer size is the amount of data BRU will transfer from/to the archive device during each read/write operation.

This parameter is usually the one to adjust when trying to maximize BRU's speed. The optimum value varies widely, depending on your system and device. If you are having problems, start by setting bufsize=16K. This should work for almost any UNIX system. There is no single "best" value for bufsize. On some systems, bufsize=8K may produce the best results. On others, setting bufsize=128K or larger may be faster. **IMPORTANT NOTE**: Over 75% of all reported BRU problems are caused by buffer sizes that are too large. This is especially true for devices that have a SCSI interface. In many cases, these problems are not apparent when writing to a device (backing up data), but occur when attempting to read (restore data).

DO NOT USE A LARGE BUFFER SIZE SIMPLY BECAUSE IT IS FASTER!

Verify your backup! Use the –i or –d options to verify the backup, or let AUTOSCAN do it automatically. A fast backup is worthless if it contains errors.

The buffer size can also be specified on the command line with the –b option. This will override any value specified in your brutab. Listed below are additional alphanumeric brutab fields. These are optional and can usually be ignored:

maxrewindtime

The maximum number of seconds that it takes to rewind a tape. Normally, the UNIX system causes the tape to rewind and then returns control to BRU when the rewind is done. However, some tape drives (or device drivers) have a design under which they return control to BRU immediately before the rewind has finished - which causes errors when BRU attempts to read the tape during AUTOSCAN. If your tape drive has this bug, setting maxrewindtime will cause BRU to wait before attempting to start an AUTOSCAN. If necessary, a setting of maxrewindtime=180 seems to work for most tape drives.

minrewindtime=

The minimum number of seconds that it takes to rewind a tape. This parameter is only in effect only if maxrewindtime is set to a non-zero value. In most cases, minrewindtime does not need to be specified and defaults to 25 seconds.

Asbufsize=

AUTOSCAN buffer size. By default, this is same as the value specified by bufsize. Normally, this is the best value and there is no need to specify asbufsize.

maxbufsize=

Maximum I/O buffer size for this device. The maximum buffer size this device can handle. If you ask BRU to use a larger buffer size, it will issue a warning that the maximum buffer size has been exceeded. Without this warning, it can be hard to determine the cause of read or write errors that are due to exceeding the capabilities of the device.

seek=

Minimum seek resolution. Zero if no seeks allowed. For most seekable devices, the seek resolution corresponds to the physical block size of each data block on the device. All seeks performed on the device will be an integral multiple of this value.

ederr=

Error code returned by the device driver when the end-of-device is reached. This allows BRU to detect when it has reached the end of a tape. On many systems, this should be set to ENOSPC. For example:

          ederr=ENOSPC

Device drivers vary widely, so the correct value for your system may be different. If you do not know the error code that is returned, do not specify ederr or set ederr=0.

prerr=

Error code returned by the device driver for partial reads. A partial read successfully reads more than zero bytes but less than the requested number of bytes.

pwerr=

Error code returned by the device driver for partial writes. A partial write successfully writes more than zero bytes but less than the requested number of bytes.

zrerr=

Error code returned by the device driver for zero length reads. A zero length read reads zero bytes.

zwerr=

Error code returned by device driver for zero length writes. A zero length write is one that writes zero bytes.

frerr=

Error code returned by device driver after an attempt to read from unformatted media. Applies mostly to floppy drives (most tapes are not formatted).

fwerr=

Error code returned by the device driver after an attempt to write to unformatted media. This code applies only to devices that must be formatted (like floppies or mini-cartridge tapes). It does not apply to most cartridge tapes, as they do not require formatting.

wperr=
>   Error code returned by the device driver after an attempt to write to media that are write-protected.

fmtcmd=
>   Allows you to specify a format command that BRU will execute if it tries to write to unformatted media (like a floppy disk that has not been formatted). The format command should be enclosed in double quotes. For example, the following command formats a floppy disk under SCO OpenServer:

>   `fmtcmd="format -f /dev/rfd096"`

>   If fmtcmd is specified, the format field must also be specified or the fmtcmd string will not be executed.

iotimeout=
>   Read/write timeout value. This is not a fixed timeout but is the number of seconds added to the maximum time needed to read/write a block of data at 1000 bytes/second. For example, the time needed to write 20,000 bytes would be 20 seconds. If iotimeout=30 is specified, this value would be added to 20. In this case, BRU will issue a timeout error if the write did not complete within 50 seconds. Normally, iotimeout can be left out. It is useful for trouble-shooting purposes if BRU "hangs" mysteriously when writing to a device.

openretries=
>   Number of additional times BRU will try to open a device if the first attempt fails. The default value is 1. Specify openretries=0 if you want only one attempt at opening a device.

opentimeout=
>   Number of seconds BRU will wait while attempting to open a device. The default value is 3600 seconds. This value must be large enough to include the amount of time it takes to rewind a device, since many devices do not return from an open call until the rewind is complete. Specify opentimeout=0 to disable the timeout.

The following parameters apply only if double buffering (the –D option) is used. If double buffering is not used, they are ignored.

shmmax=
>   Limit on the size of each shared memory segment. BRU will not attempt to allocate a shared memory segment larger than this limit.

shmseg=
>   Limit on the number of shared memory segments. BRU will not attempt to allocate more than this number of shared memory segments. For double buffering, BRU allocates one segment for shared variables, and then at least one additional segment for shared I/O buffers. Therefore, the minimum number of shared memory segments is two.

shmall=
>   Limit on the total amount of shared memory used. BRU will not attempt to allocate more than this total amount of shared memory.

Boolean brutab Fields

>   This section contains a listing of the fields with Boolean capabilities currently recognized by BRU. These Boolean fields are:

ignoreclose
>   Ignore the error that is generated by some devices when they are closed. Some devices with a SCSI interface may need to have this field specified. This will eliminate the [W003] warning message that occurs with some tape drives.

noautoscan
>   Disable the AUTOSCAN feature. The AUTOSCAN feature acts as an early-warning system - detecting minor problems before they become serious. If AUTOSCAN is disabled, your data integrity may be compromised and many types of problems can go undetected.

noreopen

Do not close and reopen archive upon media switch. This setting is rarely used but keeping the archive device open usually prevents other processes from accessing the same device.

rawtape

Archive device is a raw tape drive. This means that the kernel does not buffer data to and from the device.

norewind

Archive device does not automatically rewind to the start of the media after it has finished writing. Note that the size parameter should be zero. **NOTE**: AUTOSCAN will be disabled if norewind is specified.

advance

Indicates that the device has the capability of advancing the media past read/write errors. In our experience, very few tape drives (or device drivers) have this capability. Most refuse to advance past hard errors on the media. If this parameter is set, BRU will be able to proceed past bad spots on the media. **NOTE**: It is a serious mistake to define this parameter for devices which are unable to advance the tape drive when an error occurs. If defined incorrectly BRU may not be able to recover from read errors. Most tape drives do not have the ability to advance on error.

qfwrite

For the first write to the first media in this device, request confirmation to proceed. This flag should be used if the device is also used for mounted filesystems, to protect against accidentally overwriting a media that may have been left in the device (floppy drives for example). **NOTE**: Does not work when multiple archive devices are specified.

shmcopy

Indicates that the device driver for this device cannot do I/O directly from shared memory. During archive writes, the data will be copied from shared memory to a locally allocated buffer before doing I/O to the archive device. During archive reads, the data will first be read into a locally allocated buffer, and then copied to shared memory. BRU will warn you when it detects conditions that indicate that this parameter should be set, and will automatically switch to copy mode.

NOTE: This parameter is needed only with double buffering (–D option). If double buffering is not used, this parameter is not needed.

Global brutab Parameters

Several BRU parameters are not specific to a device. These are global parameters and are specified in the brutab file in a form that is different from device parameters. Global brutab parameters are specified with names that are all **UPPER CASE**. Only one parameter is allowed per line. A line containing a global parameter must begin with the two-character sequence #+ (the pound sign followed immediately by a plus sign).

Normally, the global parameters are specified at the beginning of the /etc/brutab file. Global brutab parameters can also be set as environment variables. These environment variables will override the settings in the /etc/brutab file. To remove any of these environment variables use the unset PARAMETER command i.e. unset BRURAW.

Here is a list of global brutab parameters (with default values), showing how they are specified:

```
#+ OVERWRITEPROTECT=YES
#+ RECYCLEDAYS=7
#+ MAXWRITES=200
#+ ZBUFSIZE=500k
```

These parameters are explained in greater detail:

SHELL=/bin/sh

This variable points to your command shell. This shell is used when BRU spawns a task for running other tasks (such as switching tapes in a loader.)

OVERWRITEPROTECT=yes

BRU will check the tape (by reading the first block) before attempting to do a backup. BRU will read the date of the tape before proceeding. BRU will then compare the date in the archive header of the tape to the number of days specified in the RECYCLEDAYS filed. If the tape is newer then the number of days specified by RECYCLEDAYS field BRU will prompt you for a different tape. **NOTE**: This parameter only take effect for devices

that have been specified and defined in the `/etc/brutab` file. They have no effect for other devices.

RECYCLEDAYS=7

The values specified by RECYCLEDAYS field is read by BRU and compared to the tape. Here's an example: Assume that you are doing today's backup and accidentally try to use the backup tape from yesterday. If RECYCLEDAYS=7 is set, then BRU will give you a warning and refuse to overwrite yesterday's tape. To continue, you will need to use a blank tape or a BRU tape more than a week old. You should set RECYCLEDAYS to a value that is consistent with your tape rotation schedule.

You can disable date checking by setting RECYCLEDAYS=0. However, this will not completely disable OVERWRITEPROTECT. BRU will still do a "tape change" check on multi-volume archives. **NOTE**: This parameter only take effect for devices that have been specified and defined in the `/etc/brutab` file. They have no effect for other devices.

MAXWRITES=200

The MAXWRITES option works in conjunction with the OVERWRITEPROTECT option and allows you to limit the number of times a specific piece of media (tape) can be used before it is recommended for removal from tape rotation. If this value is exceeded, BRU will issue a warning and refuse to continue. Below you will find a few recommendations for various tape types. Although some of these settings might seem low compared to the manufactures specifications we feel that a more conservative value will insure the integrity of your data.

4MM/8MM DAT	200	
AIT 1/2/3/4	500	
VXA-1/2		500
DLT		1000
SDLT	1000	
SAIT	1000	
LTO		10000

Setting MAXWRITES=0 will disable checking of the number of writes. **NOTE**: This parameter only take effect for devices that have been specified and defined in the `/etc/brutab` file. They have no effect for other devices. The RECYCLEDAYS and MAXWRITES parameters have no effect unless OVERWRITEPROTECT is specified.

BRUTABONLY=no

If BRUTABONLY is specified as a global parameter, BRU will permit only the use of devices listed in the `/etc/brutab` file. If the user attempts to use a device that is not listed, BRU will issue a warning message and refuse to continue until the correct device is specified.

DEVNAMECHECK=no

DEVNAMECHECK causes BRU to perform a "sanity check" on the device name to prevent the creation of a large file in the `/dev` directory. If BRUTABONLY is YES, then this option is inactive.

MATCHLEVEL=2

MATCHLEVEL determines the level of pathnames matching performed by BRU during a backup or restore. A level of 0 will only match the exact pathname entered (i.e.: `/tmp/123` will not match `./tmp/123` or `tmp/123`, only `/tmp/123` will match). A level of 1 will match relative pathnames that indicate the same path (i.e.: `tmp/123` will match itself and `./tmp/123`, but `/tmp/123` will not match). Level 3 will match any similar pathname (i.e.: `tmp/123` will match `/tmp/123`, `tmp/123` and `./tmp/123`). **NOTE**: This setting can cause some confusion, be aware of what your MATCHLEVEL is set to when doing restores. By default we have set this option to two (2) to allow maximum flexibility when restoring. This is covered in more detail in Chapter 17, "Extracting Files: The BRU Restore Function"

MAXFILENAMELEN=255

This option allows you to transfer BRU archive files from a UNIX system that supports long file names (greater than 14 characters) to a system with short file names. If you set MAXFILENAMELEN to the maximum length your system will support, it causes files with longer names to be renamed (not truncated). A warning message is issued for each renamed file. The default setting for MAXFILENAMELEN is the maximum file name length your system supports. On certain versions of UNIX (usually older ones), MAXFILENAMELEN defaults to 14. On newer OS's (like Solaris and Linux), it is set to 1023 or more.

READCHECKLEVEL=1

On many NFS mounted volumes, it isn't possible for BRU to easily determine if a file is locked. To prevent excessive read errors or a possible hang condition, BRU will pre-read files to check for a lock state. A 0 means no pre-read, a 1 means pre-read only files that appear to be locked and a 2 means to pre-read ALL files (which can slow your backup performance).

ZBUFSIZE=1000k

If most of your files are small, (less than a megabyte in size), then a setting of ZBUFSIZE=500K (or smaller) would probably work well. If you have many large database files (greater than 10 megabytes), then a setting of ZBUFSIZE=5M (or larger) might work better. For example, if BRU compresses a 500K byte file by 50%, then it would need 250K bytes to store the compressed data in its compression buffer. To compress a 2 megabyte file by 50%, 1 megabyte would be needed, etc. NOTE: Setting ZBUFSIZE to a large value may cause BRU to fail. This is not a problem with BRU but is due to limits on the amount of memory that can be allocated to a UNIX process. You may be able to increase the maximum available process memory by adjusting your UNIX kernel parameters but this is not recommended unless you are familiar with "kernel-tuning" and know what you're doing.

BRUHELP=/bru/bruhelp

This variable points to the location and name of the BRU readme or helpfile. By default the location of that file is /bru/bruhelp.

BRUMAXWARNINGS=1000

This variable sets the maximum number of warning that BRU will report before aborting the current operation. By default this is set to 1000.

BRUMAXERRORS=500

This is the variable that sets the maximum number of errors that BRU will report before aborting the current operation. By default this is set to 500.

BRUXPAT=/Library/Application Support/ArGest Backup/etc/bruxpat

This variable sets the location of the bru exclusion file. Paths and files listed in this file will be excluded from the backup or restore if the **-QX** option is placed in the bru command line, refer to Chapter 15, "File Inclusion and Exclusion." By default the file location and name is **/Library/Application Support/ArGest Backup/etc/bruxpat**.

BRURAW=/etc/bruraw

This variable sets the location of the file containing descriptions of raw data partitions to be backed up or restored by BRU. By default, this files location is /etc/bruraw. This file and the commands used with bru to activate this file are in Chapter 5, "Archive Creation" later in this manual. This used in conjunction with the –r option.

BRUSMARTREST=/Library/Application Support/ArGest Backup/etc/brusmartrest

This variable sets the name and location of the BRUSMARTREST file. Files listed in this file will be handled as open and restored using a protected method, alleviating the problem of restoring over an open file or shared library. By default this file is located in **/Library/Application Support/ArGest Backup/etc/brusmartrest**.

BRUREMOVELOG=/bru/bruremovelog

This variable sets the location and name of the results of the SmartRestore. In the event that a restored file's "text busy" flag was set, we rename the original file and then restore the appropriate file. As a result, we create a Bourne shell script that will clean up the old, renamed files. If you are creating scripts to run

BRU, it is a good idea to execute this file as the last stage of any restore that is preformed. By default this file is located in `/bru/bruremovelog`

`MOUNTCMD="(null)" /bin/mountcmd`

This parameter allows the user to specify external commands to handle devices. MOUNTCMD is used to specify a command that will be called before BRU attempts to open a device for reading or writing. **NOTE**: If you set this option with a environment variable and want to "unset" it, then use the following command "unset MOUNTCMD" (under the Bourne shell).

`UNMOUNTCMD="(null)" /bin/unmountcmd`

The UNMOUNTCMD should be used to specify a command that will be called after BRU has finished reading/writing. **NOTE**: If you set this option with a environment variable and want to "unset" it, then the following command "unset UNMOUNTCMD" (under the Bourne shell). The MOUNTCMD and UNMOUNTCMD parameters were designed for use with multiple-cartridge tape handlers (stackers or jukeboxes). BRU passes four arguments to the commands specified by MOUNTCMD and UNMOUNTCMD.

Argument	Description
1	device name (i.e. ntape0)
2	volume number (i.e. 2)
3	mode letter (i.e. 'c' or 'x')
4	media size in Kilobytes (i.e. 150000)

In most cases, the commands specified will be shell scripts. Normally, these scripts issue commands that will cause a tape handler to load or unload a tape. See Appendix K, "Using MOUNTCMD and UNMOUNTCMD," for samples of shell scripts that use these commands.

`NOFILESINHDRS=no`

With this option your backups are backwardly compatible with older versions of BRU prior to 14.2.

`DONTLABELALLVOLS=no`

This option toggles BRU's ability to write a Label for each Volume in a multi-volume backup. By default BRU will create a label on each volume during its backup process.

`DIRDATESELECT, ZINBUFFER & ZOUTBUFFER`

NOTE: For Technical Support Use Only. Please leave these settings unchanged.

Determining Parameter Values: –hh

BRU may be used with the –hh option to print a list of all parameters in the brutab file and their values. The –hh option may also be used with a device name to print information for that device from the brutab file. This allows you to determine how BRU is interpreting parameters. See Appendix B, "The BRU Help Command" for sample command lines and output.

<u>14</u>
Terminal Archive Creation:
The Backup Function

Overview

This chapter describes command line options that are used to create a new archive - BRU's backup function - and contains sample commands for the basic types of backup. The options that can be used with the bru command allow you to tell BRU:

- You want to perform a backup
- To estimate how many tapes or disks it will take to create the archive
- What device to use
- How to label the archive
- To set the verbosity level for BRU
- To generate log files
- To wait for confirmation for each action it takes
- Not to cross local filesystem boundaries
- Not to cross boundaries between local and remote filesystems
- To run without user intervention
- To back up a set of files or directories based on a list supplied on the command line
- To back up a set of files by modification date
- To back up a set of files owned by a particular user
- To back up a set of files or directories based on a list supplied in a file or read from stdin

NOTE: In this section we talk about writing archives or archiving files. If you are unfamiliar with this way of referring to the backup function, see the "Definitions" section of Chapter 1.

The BRU Command Line

The generalized BRU command line looks like this:

```
bru modes [control options] [selection options] [files]
```

where bru is the command and modes tells BRU, for example, whether you are backing up or restoring files. The most commonly used control options specify the device on which an archive is written, the size of the buffer to be used, or the string with which the archive is to be labeled. The selection options tell BRU which files or directories to archive. These might be files owned by a particular user or files created or modified after a particular date.

Options are always start with a hyphen and a letter that indicates the type of option. Some options also require parameters. The following sections describe how each of the most common options is used. See Appendix C, "The BRU Manual Page," for a detailed description of all options.

Command Line Options: Modes

Modes tell BRU what to do, for example, create an archive, list the contents of an archive, or extract files from an archive. There must be at least one mode option on the BRU command line. This chapter is concerned with describing BRU in archive creation mode.

Telling BRU This Is a Backup: -c

The −c option creates the archive. It does not require a parameter. All uses of bru to create archives will begin with

```
bru -c
```

The command line shown above is the simplest backup command you can give. But it assumes two things: First, since it does not specify a device, it assumes that your /etc/brutab file has been set up to specify a default device (see "Telling BRU What Device To Use: −f," below; Chapter 13, "Setting Device Parameters: The brutab File"; and Appendix E, "A Sample brutab File"). Second, it assumes you want to use BRU's default value for files. When writing an archive, the default for BRU is to select all files in the current directory and any subdirectories of the current directory.

Estimating Archive Size: −e

Estimating the size of the archive you want to write is similar to creating the archive. If you use the −e option instead of −c, BRU scans the files you want to back up and tries to tell you how many tapes (or disks if you are backing up to disk) will be needed to create the archive. The estimate is based on the size of the archive media you have either specified on the command line using the −s option, or implicitly specified by selecting a device named in the brutab file. See "Specifying Media Size: −s," below.

The resulting number will be accurate as of the time BRU is run. However, if the file system is active, changes in the files may cause the number of volumes to change. In addition, if the last file to be archived requires one more volume, that information may not be taken into account in the estimate.
Estimate mode cannot be used with data compression since BRU would have to compress each file to find out how much space it would take up in an archive. This would probably take nearly as long as creating the archive. If you must have an estimate that uses file compression, you can create an archive and send it to the null device:

```
$ bru -cvvvv -Z -s 640K -f /dev/null [files]
```

The above command will write a compressed archive to the null device and the verbosity output will indicate how many 640K volumes are needed to save the archive. See "Data Compression: −Z," below.

Command Line Options: Control Options

Modes tell BRU what you want to do; control options tell BRU how you want it done. Control options, as the name implies are optional. If you do not provide them, system defaults are used. Either "reasonable" defaults are used or default values are taken from the /etc/brutab file, which you will have customized for your system. (Chapter 13, "Setting Device Parameters: The brutab File," shows you how to set up a brutab file.) Control options that are commonly included in the /etc/brutab file are the device on which to create the archive and the size of the archive media.

Telling BRU What Device to Use: −f device

If you do not specify a device on the command line, BRU defaults to the first entry in the /etc/brutab file. See Chapter 13, "Setting Device Parameters: The brutab File."

The example below shows how to specify the device on the command line. It creates an archive from the files in the /usr/src directory, using a 20 gigabyte tape inserted into the drive mounted as ntape0. It is important to know that the file that follows the −f need not be a device file, although this is the usual case. You may also use files on the system as archiving device files. However, you will experience an error if you set the BRUTABONLY=YES global brutab parameter in the brutab file.

```
cd /usr/src
bru -c -f ntape0 -s 20GT
```

See "Specifying Media Size: –s," below.

Labeling the Archive: –L label

The –L option lets you provide a label of up to 63 characters for your archive. There are two ways of providing this label to BRU. The first is to simply specify the label from the command line as shown below.

```
bru -cv -L "Test Label" /usr/bin
bru -c -L "Old Mail 1" -vvvv letter* memo*
```

The second method of passing a label string to BRU, is to create a file and tell BRU to read that file. BRU will read in the first 63 characters of the file as the label for the tape as shown below.

```
bru -c -L "/tmp/label name" /usr/bin
```

The label information appears in the archive header, which can be viewed (or fed to an archive manager program) using BRU's –g option. **NOTE**: That the label string and file name are enclosed in a pair of double quotes. This assures that any shell program you might be running (sh, bash, csh, ksh, tsh, zsh) will treat the label as a single parameter. If the label is not enclosed in double quotes, BRU will see the individual components of the label string as separate parameters. In the second case, Old would become the label, and each of the words in the rest of the label would be treated as a file name. The following shows how –g mode is used to read the label information:

```
bru -g -f oldmail
```

In this example, oldmail is a standard UNIX file we are using in place of a device to receive the archived mail files. See Chapter 16, "Archive Inspection and Verification," and Appendix C, "The BRU Manual Page" for a description of the –g option.

Backing up Raw devices: –r

This enables BRU to backup or restore raw data partitions. A Raw data partition is sometimes used by large database programs for storage. These partitions can now be backed up with BRU as long as a file

called /etc/bruraw is created with the definitions of that device included in it. The file should contain entries in the format below :

Raw Device name	Size	Blk Size	Starting Offset
/dev/rdisk0	30G	512	0

NOTE: An entry **MUST** exist in the bruraw file or BRU will abort the operation with an error message.

When backing up raw partitions, you must either have the device node in the tree relative to your current location or you must explicitly declare the raw device on the command line as a file name. For example, if you are in / and the raw device node in /dev/rdisk0, you can use this command line to back it up properly:

```
bru -cv -r /dev/rdisk0
```

If, however, you are invoking the bru command from your home directory, you must include the path to the device node on the command line:

```
bru -cv -r /dev/rdisk0 /dev/rdisk0
```

NOTE: If any writes occur during the backup, we will issue error messages explaining that some data changed during the backup. We do suggest that if this happens, you should run your backup process again and attempt to get a clean image of that data. BRU starts reading at the front of a file and works its way through it. We do not lock files so if the information was changed during our backup that information on the tape or archive device will be incorrect.

Changing Default BRU Operations With: –Q

This option allows you to change or reset BRU default operations set in the BRUTAB file. By using the –Q option with one of it's create mode modifiers (H or L) BRU will disable that setting and allow you to run with the new settings.

-QH - This option will disable BRU from placing small files into tape headers. Using this option allows you to create BRU tapes that are compatible with versions of BRU prior to 14.2. This will override the global BRUTAB setting of NOFILESINHDRS.

-QL - This option tells BRU to use the literal string as a tape label. This overrides BRU's attempt to look for a file from which to read the tape label. This can be useful if you have a file (in the directory that you are running BRU from) that has the same name as your label string.

-QV - This option is used when extracting (restoring) and tells BRU to ignore that fact that the operation is not starting on the first tape of an archive set and not prompt with a warning.

-QX - This option enables the use of the include/exclude patterns file. The default location for the file in ArGest Backup on Mac is:

```
/Library/Application Support/ArGest Backup/etc/bruxpat
```

Selection Depth for Backup –U

BRU, by default will expand directories when given from the command line. With this option BRU can be limited to a certain level of directories that it will expand. By running BRU with the –U0 (zero) option BRU will only backup the first level of the directory given or the current directory. This can be very useful when you need to backup only the files from a specific directory and none of the files below that directory. For example, the command below will only backup the current directory depth. BRU will backup all files in /home/web and all directory nodes or (names) but it will not pass through and backup files in other directories:

```
bru -c -V -U0 /home/web
```

Below you will see an example of running the option but with the –U option set to two (2):

```
bru -c -V -U2 /home/web
```

With this folder depth setting, BRU would backup all files and nodes in the /home/web (level 0) directory along with /home/web/test (level 1), /home/web/test/runone (level 2), but BRU would not backup files in the /home/web/test/runone/demo1 (3 levels deep).

Setting the Verbosity Level: –v and –V

BRU normally does its work silently, returning control to the calling interface (the user terminal, or the spawning process) once it has completed its task. However, if you want to monitor what BRU is doing, a verbosity level selection option is accessible from the command line. Verbosity is specified by the –v (lower case) and –V (upper case) options. The verbosity options provide five levels of verbosity and a summary mode: –v, –vv, –vvv, –vvvv, –vvvvv, and –V, where a single v represents the lowest and five v's the highest verbosity level. Using –V produces an execution summary without the volume of output generated by the other verbose options. The following example shows how the verbosity option is used with the estimate mode:

```
$ cd /
$ bru -e -vv ./bin/a*
e 2k of 4k [1] ./
e 2k of 6k [1] ./bin
e 72k of 78k [1] ./bin/acctcom
e 68k of 146k [1] ./bin/adb
```

The first field in the output contains the mode character, as a reminder of what mode is currently running (in this case the "e", or "estimate" mode). The next field gives the amount of space in kilobytes that this particular file will use in the archive. The next field ("of XXXk") gives a running total of the amount of data in the archive, including the current file. The number inside the square brackets is the current volume number. Finally, the name of the file is listed.

Generating Log Files Using −v (lower case v)

In the previous section, we monitored the output produced by the −v option on the screen. The −v option also allows you to generate log files that contain this information. If you want to keep a record of how a particular tape archive was created and what files it contains, you can redirect the output of one of the verbose options to a log file. This command line sends the information produced by the −v option to the screen:

```
bru -cv file1 file2 file3
```

but this command line sends the log information to logfile and any error messages to a different file, errfile:

```
bru -cv file1 file2 file3 > logfile 2> errfile
```

If you want both error and log information to go to the same file, you can use a command line similar to this under the Bourne shell:

```
bru -c -v file1 file2 file3 > logfile 2>&1
```

If you want BRU to read the list of files to be archived from a file instead of from the command line, you can use something like:

```
bru -c -vvvv < filelist > logfile
```

where filelist is a file containing the list of files, one to a line.

The amount of information you collect in your log files can be varied by using different levels of verbosity.

All error messages and warnings are also recorded in the BRU execution log, /var/log/bruexeclog. The bruexeclog file is described in Appendix F, "BRU's Execution Log."

BRU offers an additional verbosity level that is designed to allow for easier parsing within a script or interface. By setting the verbosity level to 10 v's, the output from BRU will appear as:

```
VL:c|e10|1|76|81a4|0|0|3e3199d3|3e10f6ac|3e10f6ac|0|70|
803||||/etc/mtab
```

In this output, the interesting fields are the 3rd and 12th fields separated by the pipe (|) symbol. The 3rd field tells you what volume (if it is a multi-volume backup), while the 12th field tells you the logical block address within the volume. In the example above, the /etc/mtab file is located on the first volume at the 0x70 (HEX value) block.

Confirmation Option: −w

If you are not quite sure what BRU is going to do, you may want to use this option as a test, on a small subset of files. When the −w option (wait for confirmation) is specified, for each file, BRU:

- Prints the file name
- Shows the action it is about to take
- Asks for confirmation

If you have a very long list of files, the confirmation option can be tedious. BRU therefore provides a special response that lets you change your mind. If you type a g (for "go ahead") instead of y or n, BRU will stop asking for confirmation. The following example shows how this response is used.

```
$ bru -cvw -f /dev/null ./bin
c ./: please confirm [y/n/g] y
c 2k of 4k [1] ./
c ./bin: please confirm [y/n/g] y
c 2k of 6k [1] ./bin
c ./bin/acctcom: please confirm [y/n/g] y
c 72k of 78k [1] ./bin/acctcom
c ./bin/adb: please confirm [y/n/g] y
c 68k of 146k [1] ./bin/adb
c ./bin/ar: please confirm [y/n/g] g
c 36k of 182k [1] ./bin/ar
c 120k of 302k [1] ./bin/as
c 16k of 318k [1] ./bin/asa
c 4k of 322k [1] ./bin/basename
   .  .  .
```

IMPORTANT NOTE: Be aware that once you have typed g in response to BRU's request for confirmation, you cannot go back to confirmation mode.

Do Not Cross Mount Points: −m

There are times when you may want to back up all files in a single directory, but you may also want to limit the selection of files to those on the same mounted filesystem as one of the files in the files parameter. You can do this by including the -m option on the command line when

you create the archive. For example, this command line will back up only files in the root directory and exclude any files on mounted file systems:

```
bru -c -v -m /
```

The next example saves all files on both the root and the /usr filesystems, but none of the files from other filesystems, such as /usr2, is included:

```
bru -c -v -m / /usr
```

If you are not certain which file systems are currently mounted, you can use the df command to list this information. The output of df varies from one UNIX system to another, but will generally list one filesystem per output line.

Excluding Remote Files: −R

Specifying the −R option tells BRU to exclude all files on remote systems. It lets you back up files on your local system and not reach out across the network to other systems. The following example backs up only files on the local system:

```
bru -cv -R /
```

For an individual machine that has remote connections and mounted filesystems from remote machines, archiving files from the remote machines could mean generating a very large archive. The −R option should be used to limit the backup to your local files.

Running Without User Intervention: −B

The −B option is used when running BRU in the background. Normally, it is included as part of a shell script that is scheduled to run automatically. For example, the following line might be included in a script that is run by cron every night:

```
bru -c -B /usr
```

The −B option in the BRU command line tells BRU that it is running in the background and that it can expect no user intervention. It is specifically designed to prevent the situation in which a background backup process needs another tape, and because there is no one present to mount it, the process waits indefinitely for input. If −B is used, the process will terminate with an appropriate error code. **NOTE**: The −B option will be

set automatically if BRU is started as a background process, that is, if the "&" shell option is included on the command line. For example:

```
bru -c /usr &
```

This is true when using the Bourne shell; it may not work for other shells.

If an operator is available, you may decide not to use the −B option in order to assure that the backup actually takes place. If −B is not specified, then BRU will indeed ask for help and the operator may be able to resolve the error condition for a successful run.

The −I option provides other options for communicating with BRU in the background. See Chapter 9, "Advanced Uses," "Using BRU with brutalk: −I," Appendix C, "The BRU Manual Page," and Appendix H, "The brutalk Program." The selection of −B or −I will depend on what the desired recovery is when interaction is necessary.

Specifying Media Size: −s media_size

The −s option allows you to specify media size in the following format:

```
bru -c -s msize [options] [files]
```

msize is usually expressed as kilobytes (K bytes), megabytes (M bytes), gigabytes, or occasionally as bytes. Normally this option is only required with −c mode since BRU writes this information onto the archive header block. If specified, msize overrides any existing default value, whether built in or read from the archive header.

The following command is an example that writes to a 3.5" floppy disk using a size of 1440KB. It will back up all the files in the /tmp directory:

```
bru -cv -s 1440K -f /dev/rfd0135ds18 /tmp
```

Data Compression: −Z

Software based data compression is specified using the −Z option. In addition to allowing more data to be stored in a smaller space, using compression in software can provide a performance improvement if the backup device used does not offer hardware compression. However, if you are writing to a locally attached backup device that does offer hardware compression, do not use BRU's software compression.

Hardware compression will always be more efficient than software compression and sending compressed data to a compression-enabled backup device can result in the backup becoming larger, rather than smaller.

Command Line Options: File Selection

The files parameter on the command line tells BRU which directories and files to include in an archive. BRU scans its command line, and when it runs out of mode, control, and selection option specifications, interprets the rest of the command line as file names. File names must be specified last. If they are not, the results of the command may be unexpected. It is important to understand that specifying files on the command line is optional. If you do not include the files parameter, BRU will use the default value and include everything in the current directory and in all directories below the current directory.

Using Wildcard Filename Expansion

When you are executing the bru command from the UNIX command line, you can specify the files parameter by including wildcards (special characters that the shell expands into matching filenames). BRU recognizes wildcard strings in the same format recognized by the Bourne and Korn shells. For example, if you have the following files in a directory:

```
file1
file2
file3
```

and if you enter this command on the command line (not, for example, as part of an executable shell):

```
bru -cv file*
```

The wildcard character "*" of the above command will be expanded by the UNIX shell and will match the three files. The command is equivalent to

```
bru -cv file1 file2 file3
```

This expansion is done by the UNIX shell, not by BRU.

Wildcard file name expansion can cause problems, especially if the command is entered from a directory in which there are many files or subdirectories. The following command is NOT necessary and is NOT recommended:

```
bru -cv *
```

Using the above syntax can result in your shell attempting to expand the wildcard character and could result in a command line buffer overflow. While BRU will not allow this to become a potential system security exploit, it will most likely cause the BRU command to fail, rather than run. If no files are specified, BRU will automatically back up all files in the current directory. The above command should be replaced with:

```
bru -cv
```

If you have problems when using wildcard names, it may be due to filename expansion limits set by the UNIX shell. To see how the filenames were expanded, check the BRU execution log, `/var/log/bruexeclog`. The "START" entry should contain a list of all the expanded names that were passed to BRU. **NOTE**: Wildcard characters are used in a slightly different manner when restoring from an archive. Wildcard characters used to specify file names with −x (file extraction) must be enclosed in double quotes. See Chapter 8, "Extracting Files: The BRU Restore Function," "Using Wildcard Filename Expansion."

Using Pathnames

Files may be specified using either relative pathnames or absolute pathnames. The default used by BRU is to save each file by specifying the path to the file as relative to the current directory. This takes the form:

```
./subdir/filename
```

Naming a file using its relative pathname allows you to transport it from one filesystem to another because the path is relative to the current directory. For example, files that came from `./usr/tests` on one machine may be restored to `./system/qa` on another machine without the need to tell BRU anything about either the source or the target directory (since everything is relative to the current directory).

See Chapter 8, "Extracting Files: The BRU Restore Function," "Where To Put the Files: Pathnames," for a note on specifying file names. If files are specified, the names as stored will no longer be specified as relative to the current directory unless you explicitly include "./" in your file specifier(s). If you want your files to be portable from one machine (or filesystem) to another, you may want to specify files using a relative pathname rather than an absolute pathname. If you have backed up your files using absolute pathnames, BRU has an option (−PA) that allows you to convert an absolute pathname to a relative pathname when restoring the files. See the section on "Converting Pathnames from Absolute to Relative Form: −PA, " in Chapter 8, "Extracting Files: The BRU Restore

Function. **NOTE**: The string that describes the path to a file must not exceed the MAXPATH value for the shell within which BRU is being executed. For normal Bourne Shell, `/bin/sh`, this is usually 1023 characters.

Selecting Files from `stdin`: −

As an alternative to specifying the files to be saved on the command line, you can provide BRU with a file that it can read, or you can pipe the output of a command to BRU and tell BRU to interpret this output as the file list. In both cases, you must use the special filename "−". When you use a hyphen on the BRU command line in place of the files option, BRU reads the standard input stream for a list of files to process rather than collecting them from the command line. This usage takes the form:

```
bru -c [options] -
```

The special filename "−" is typically used in conjunction with a UNIX pipe or with the standard input stream redirected to a file containing a list of filenames. The two commands

```
cat /tmp/filelist | bru -c -PEPf -
```

and

```
bru -c -PEPf - < /tmp/filelist
```

are equivalent. Both create an archive of the files you have listed in `filelist`. Note the use of the −PEPf option, which causes BRU to expand any directories in the list. If the −PEPf is not specified, BRU will NOT expand directories, but will simply archive a single entry for each directory (just the directory name).

The following command backs up all files owned by a specific user, in this case user `jim`. The UNIX find command translates the symbolic user name `jim` to the correct user id. In this case, the −PEPf option is not needed, because the find command expands the directory and passes all the pathnames to BRU:

```
cd /
find . -user jim | bru -c -
```

Here is an alternate way to let BRU do the same thing. Since BRU is not reading a list from `stdin`, it will automatically expand the directories and

will back up all the files owned by `jim` in the current and any lower directories:

```
bru -c -o jim /
```

Reading from `stdin` and Writing to `stdout`

A slightly different form of the BRU command is used when reading a list of files from `stdin` and writing the archive to `stdout`. Here's an example:

```
bru -c -f - - < filelist > bru.out
```

where `filelist` contains the list of files to be saved, creates an archive with the specified files and writes it to the file `bru.out`. The first "–" is taken as the parameter for the –f option, specifying that the output is to be written to the standard output stream. The second "–" is taken as the special filename in place of the files arguments, and directs BRU to read the list of files to archive, from the standard input stream.

Selecting Files by Date: –n date

Backups you schedule on a regular basis - daily, weekly, monthly, or at whatever interval is convenient for your application - can be full backups or incremental backups. The –n option is used to perform incremental backups (the 'n' stands for newer than). It tells BRU to save only files that were created or modified after the date specified as a parameter. By default, it will cause BRU to only save files that were modified (or the `inode` status changed) after the date specified as a parameter. The default mode check the modification time and the status (`inode`) change time. If either time is greater than the specified date, the file will be selected for backup. To change the default comparison method (i.e. compare with modification time only), use the date type modifier example

```
bru -c0n 7-DEC-94,m
```

If you have a large number of files to back up, you can save time by performing incremental backups, for example, on a daily basis, and then running a weekly full backup. This way, only the files that changed since your last full backup will be archived during the week. Depending on your supply of backup tapes, you can then consider reusing an old backup tape from two or three weeks ago for a given weekly backup. Once a month you may want to do a full backup of all files and save these archives virtually indefinitely.

The frequency with which you perform backups, and how long you preserve them, depends on how far back you think users may need to retrieve earlier versions of files. This decision will depend on your system and your needs.

Date Formats for Use with –n

The date may be specified in one of four ways:

Format 1:

The variables in the first format have the following meanings:
    DD - The day of the month.
    MMM - The first three characters of the name of the month
    YY - The last two digits of the current year
    HH - The hour in 24 hour format
    MM - The minute
    SS - Seconds

The first format takes the following form:

    -n DD-MMM-YY[,HH:MM:SS][,amc]

For example,

    -n 07-dec-92,01:00

schedules a backup for 1:00 a.m. on December 7th, 1992. The three-letter month parameter is case insensitive. It could just as well have been written DEC or Dec. The seconds may be omitted. As indicated by the square brackets, the entire time specification is optional. If present, it is separated from the date specification by a single comma (,). If no time is specified, the default value, 00:00:00, is used.

There are three optional date types:

    a - Access time. Access time can be set by the touch command. The list command ls –lu shows the access time.

    m - Modification time. Modification time is shown by the ls –l command. The modification time can also be set by the touch command.

    c - Inode change time, sometimes mistakenly referred to as "create" time but also known as status change time. Modifying a file, changing permissions, or moving a file will change this time. The time is shown by ls –lc. The status change time is set by the system. This time cannot be set by any user programs.

Format 2:

The following is a second format. All components are specified numerically.

    -n MM/DD/YY[,HH:MM:SS]

Format 3:

A third format may also be used to specify the date following the –n:

    -n MMDDHHMM[YY]

In this case, the month is specified numerically (MM) and must include a leading zero for one-digit months (for example, January must be entered as 01, not as 1). Note that there are two MM fields. The first is the month, the second is the year. If the year is not specified, BRU defaults to the current year.

Here is an example that schedules a backup for February 17th, 1993 at 1:30 a.m.:

    -n 0217013093

Format 4:

The fourth option tells BRU to look for a specific file and use its modification date as the argument to –n.

    -n pathname

You can use this option to create a script that will automatically create an archive for the appropriate time interval. Thus the file name you specify may be nothing more than a dummy file whose sole purpose is to provide a pathname whose modification time is controllable. **NOTE**: The –n option can also be used when retrieving files from an archive, that is, when using the –x mode. In this case, it tells BRU to extract only files that are newer than the date specified. See Chapter 7, "Extracting Files," "Selecting Files by Date: –n," for sample command lines.

Resetting Access Time: –a

The option –a causes BRU to reset the access time of the files processed. This is important when using the atime (access time) of a

file for date processing. Since the action of reading a file for backup accesses it, telling BRU to reset the `atime` will change the file's `atime` to the last real access, rather than the access just performed by BRU.

Inverse Date Specification

In rare cases you may need to back up files older than a given date. You can do this by placing an exclamation point (!) immediately before the date. The exclamation point reverses the sense of the search. For example,

```
bru -c -n !10-Jan-92
```

will back up all files older than January 10th, 1992. Be aware that you may have to use your shell's escape character to use the ! on the command line.

Selecting Files by User: −o

You may want to back up files owned by a particular user. The −o option lets you tell BRU the identity of the user whose files you want to archive. BRU accepts three different forms of user identification:

- User name
- Numeric user ID
- File owner

Selecting Files by User Name: −o login_name

The user name is an ASCII string that identifies the user. It must correspond to a user name in a password file (normally in `/etc/passwd`). The first entry on each line in the `/etc/passwd` file is the user name. Note that this is the user's login name, not the full name of the user. A typical entry is:

```
jeff:yrrZaI78sqbro:8:2:Jeff Dough:...
```

This option has the form:

```
-o login_name
```

In the above entry, jeff is the user name you would specify. It is also the information that appears as the file owner when an ls -l command is issued. The following is an example of selecting the files to be archived by user name:

```
bru -c -o jeff files
```

This command will create an archive containing only files owned by user jeff.

Selecting Files by File Owner: −o pathname

If you use a pathname to a file as the argument to the −o option, BRU assumes that the owner of that file is the user whose files are to be archived. This option has the form:

```
-o pathname
```

An example is:

```
bru -c -o /usr/dough files
```

BRU selects all files in files that are owned by the owner of the file `/usr/dough`.

Selecting Files by User ID: −o decimal_value

You can also specify the owner by his or her decimal identifier. The decimal identifier is the `uid` value in the `/etc/passwd` file. This option has the form:

```
-o decimal_value
```

An example is:

```
bru -x -o14720
```

The numeric user ID is used less often than the other two forms of the -o option, but is available for experienced UNIX users who want to use it. In some circumstances, it is possible for files to have user id numbers that do not match any user listed in the `/etc/passwd` file.

Forcing Overwrite of Archives (OVERWRITEPROTECT): −O

If you have the **OVERWRITEPROTECT** and **RECYCLEDAYS** set in the BRUTAB file, archives will not be overwritten unless their creation age is older than the setting for **RECYCLEDAYS**. To override this from the command line, use the −o (capital oh) option.

The Shell Scripts fullbru and incbru

Two shell scripts, **fullbru** and **incbru**, have been included with BRU to help you perform backups. These are normally installed as **/bin/ fullbru** and **/bin/incbru**. The **fullbru** script will perform a full backup. It will back up all files on your system. **incbru** will perform an incremental backup; it will back up only the files that have changed since the last time **fullbru** was run. The **incbru** script does most of the work. It was written to be as simple as possible and may not work correctly on your system. Feel free to modify it to suit your needs. See *Appendix G, "The Shell Scripts* **fullbru** *and* **incbru***"* for copies of the scripts and examples of their use.

Interrupting BRU

BRU normally catches both interrupt (**SIGINT**) and quit (**SIGQUIT**) signals. The keys that generate these signals are system dependent and can be discovered with the UNIX **stty** command (**stty -a** for example). When BRU receives an interrupt (**SIGINT**) signal during archive creation or extraction, it completes its work on the current file before cleaning up and exiting. By contrast, when BRU receives a **SIGQUIT** signal during archive creation or extraction, it terminates the program immediately. If an archive is being created, it will be truncated at the point at which **SIGQUIT** is received. If a file is being extracted from an archive, the file will not be restored properly. An interrupt always leaves an archive in a consistent state for future reads. It is therefore the recommended method for stopping BRU.

The bruxpat File and the -QX Option

In addition to the standard BRU options for selecting files-by date, user, or owner - BRU lets you include or exclude files by name. This feature is enabled with the **-QX** (uppercase QX) option. The **-QX** option can be used when you are creating an archive (**-c** mode) or extracting files from an archive (**-x** mode). Additionally, the **bruxpat** file is used when using BRU's software compression to exclude specified file types from the compression pass.

When you include the –QX option on the BRU command line, BRU will apply include/exclude patterns specified in the **/etc/bruxpat** file (the default pattern file). In BRU LE, the path to the bruxpat file is **/var/lib/ bru/etc/bruxpat**. The filenames encountered by BRU as it does a backup (or restore) will be compared to the patterns. If a filename matches any of the exclude patterns, it will not be backed up (or restored). Normally patterns will be read from the **/etc/bruxpat** file. BRU will use patterns from a different file if the environment variable BRUXPAT is set to the name of the new file. For example:

```
BRUXPAT=/etc/newbruxpat
export BRUXPAT
```

The **bruxpat** file is also used to disable compression on certain files (**-z** option). If many of your files are already compressed, you can enter a line in the bruxpat file that tells BRU not to compress files whose names indicate that they are already compressed. You may, for example, want to disable compression for files with filenames that end in ".z" or ".z" or MP3 files (which are already highly compressed).

In addition to the use of the **bruxpat** file, BRU versions since 17.0 variant 1.4.0 now support the use of exclusion patterns from the command line. This is done using the **-X** (uppercase X) option followed by a shell pattern for exclusion:

```
bru -cvvf ntape0 -X "*/Caches/*" -X "*/*.tmp" /Users
```

That command would backup the /Users folder on your system while excluding any Caches folder contents and any files that end in ".`tmp`". Note that the path must be included within quotes if it contains any form of wildcard or regular expression components.

For earlier version users, please make a note that the **bruxpat** option has changed from **-x** to **-QX**. This was done to allow simpler entries on the command line for one time exclusions. Please verify any pre-existing BRU scripts and change all instances of **-x** to **-QX**.

Specifying Include/Exclude Patterns

Each line in the **bruxpat** file consists of a control field and a pattern. The pattern is separated from the control field by a space or tab (whitespace). The include/exclude patterns can be specified as shell wildcard expressions or as regular expressions (as used by commands like **grep** and **sed**). Lines beginning with the '**#**' character are treated as comments.

The control field consists of two characters. The first character specifies the type of action, the second specifies the type of pattern. The control field characters are described below:

The Control Field: Action Type Characters

i   Include this pathname if the pattern matches. The pathname is accepted and no further patterns are applied.

x   Exclude this pathname if the pattern matches. The pathname is rejected and no further patterns are applied.

z   Disable compression for pathnames that match this pattern. Applies only if compression is specified (the –Z option).

The Control Field: Pattern Type Characters

s   The pattern is a shell style wildcard pattern except that '/' characters are   not treated as special characters.

r   The pattern is a regular expression.

l   The pattern is a literal string.

The Pattern Field

The pattern may be one of the three types specified above under "Pattern Type Characters." The way in which shell wildcard patterns and regular expressions are formed can be found in the standard UNIX documentation. **NOTE**: Be aware of any trailing whitespace on ANY non-comment line. This can cause unexpected problems with BRU.

A Sample **bruxpat** File

Here is a typical bruxpat file with patterns used to include and exclude files:

```
#
# Sample bruxpat
# Not all of these are necessary and are
# provided for example ONLY!
#
# Include the C runtime startup file but
# exclude all other object
# files.
is */crt0.o
xs *.o
#
# Exclude all core files.
xs */core
xs core
#
# Exclude files and subdirectories in
# the temporary directories. Handle
# both relative and absolute
# form pathnames.
xs ./usr/tmp/*
xs /usr/tmp/*
xs ./tmp/*
xs /tmp/*
#
# Don't waste time trying to compress
# files that are already compressed
zs *.[Zz]
zs *.gz
zs *.bz2
zs *.mp3
zs *.gif
zs *.jpg
```

The patterns act as filters that are applied to pathnames currently known by BRU. A pathname currently known by BRU is any pathname included by default (the current directory and everything below the current directory) or specified on the BRU command line (either explicitly or read in from files). The –X option will not cause BRU to select or include any files that are not part of the current directory or that are not among the directories or files specified as part of the BRU command line.

Testing Include/Exclude Patterns

Filter patterns may produce unexpected results, especially if several patterns are specified. It is wise to do a test before you attempt a backup. The following command will test which files will be extracted from an archive, and will write the pathnames selected to **xtest.out** using the extract pattern(s) specified in **/etc/bruxpat** (or the file named in the environment variable **BRUXPAT**). Test your parameters without doing an actual backup by using the following BRU command:

```
bru -evvvv -QX > ctest.out
```

Running BRU in **-e** mode will not perform an actual backup; instead it will print all the selected pathnames, in this case sending them to the file named **ctest.out**. If this file contains the pathnames you expect, then it is safe to do a backup with the **-c** option substituted for **-e**:

```
bru -cvvvv -QX
```

You can test patterns used for file extraction in a similar manner by using the **-t** option. The following command will test which files will be extracted from an archive, and will write the table of contents to **xtest.out** using the extract pattern(s) specified in **/etc/bruxpat** (or the file named in the environment variable **BRUXPAT**):

```
bru -tvvvv -QX > xtest.out
```

Again, if **xtest.out** contains the pathnames you expect, it is safe to restore files, substituting **-x** for **-t**:

```
bru -xvvvv -QX
```

NOTE: To use the combination of the **-T** along with the **-QX** options, refer to Chapter 10, "Extracting Files."

<div style="text-align: right">

_16_
Command Line Archive Inspection and
Verification

</div>

Overview

This chapter shows the options BRU provides for inspecting an archive or verifying its contents. These options allow:

- Inspect an archive
- List its contents
- Compare the archive to the current contents of a filesystem
- Create and list the contents of the archive list file

See Appendix J, "Archive Inspection and Verification Examples," for examples of the commands used and the output generated.

Archive Inspection

BRU provides three types of archive verification. The first (–i, inspect mode) reads the entire archive to ensure that the data and the internal storage structures are intact. The second (–d, differences mode) reads each file in the archive and compares it with the corresponding file on the disk. The third, default, method is called AUTOSCAN and is performed automatically. BRU also provides a way to list the contents of an archive without performing any verification (–t, table of contents mode).

BRU's –g option allows you to read the archive information block. This block contains the archive label and other information about the archive.

Inspecting an Archive: –i

When you use the –i (inspect) option on the BRU command line, BRU reads the archive tape and verifies the checksums of the data written. If data or internal storage structures are inconsistent, BRU will issue warning messages. You can stack –i with –c to manually inspect the backup that is run:

```
bru -c -i
```

This will back up all files in the current directory, then rewind and do a checksum verification of the archive. This is the equivalent of BRU's

AUTOSCAN mode, and is not required unless you have disabled the AUTOSCAN feature with the –A command line option or defined noautoscan in the /etc/brutab for the specific device.

You can tell BRU to inspect an archive at any time. Unlike the –d option described later in this chapter, the files used to create the archive do not need to be on line.

Listing the Table of Contents: –t

The -t option lists the table of contents of the archive. If the verbose option (–v) is included on the command line, BRU provides a more detailed table of contents. The –v option, when used with –t, provides the equivalent of an ls –l performed on your UNIX system. The –vv option, when used with –t, provides information about hard or symbolic links, as well as file sizes and so on.

When run without any of the verbose options, the table of contents is simply the list of files in the archive:

```
$ bru -t -f bru.out
/
/bin
/bin/cat
/dev
/dev/rfloppy0
/dev/bru.q
/usr
/usr/lib
/usr/lib/news
$
```

Adding –v to the command line produces a listing of the table of contents that is similar to that produced by ls –l:

```
$ bru -tvf bru.out
drwxr-xr-x 12 root root 0 Jun 11 10:33 /
drwxr-xr-x 2 bin bin 0 Jan 29 20:49 /bin
-rwxr-xr-x 1 bin bin 9525 Nov 20 1987 /bin/cat
drwxr-xr-x 5 bin sys 0 Sep 20 16:25 /dev
prw-rw-rw- 1 root root 0 Sep 2 00:40 /dev/bru.q
drwxr-xr-x 25 root bin 0 Aug 29 15:10 /usr
drwxr-xr-x 27 bin bin 0 Sep 18 23:09 /usr/lib
lrwxrwxrwx 1 news news 0 Feb 4 00:34 /usr/lib/news
```

Dumping the Archive Information Block: –g

The –g mode prints formatted information from the archive header. This information is stored in the first block (2K) of the archive. The archive header contains information about when and where the archive was created, who created it, etc. It also contains an optional 63-character label (specified by the user who created the archive).

The following example shows how to use the –g mode to list the archive header information:

```
$ bru -g
label:         2010-10-15 /Applications/BRU UB
created:       Fri Oct 15 07:54:46 2010
artime:        1287154486
archive_id:    4cb86b367287
volume:        1
writes:        0
release:       17.1
variant:       0
bufsize:       8192
msize:         0
msize_blks:    0
serial_number: UNSERIALIZED
device:        ntape0
user:          tjones
group:         staff
system:        Darwin Tim-Jone 10.4.0 Darwin K i386
bru:           mac-osx
command_line: /bin/bru -cvvOA -QB -b 8k -R -QX -j -L
2010-10-15 /Applications/BRU UB -l -f /Users
```

The archive information block fields are described below:

> **label**:
> Label that was specified with the –L option when archive was created. Blank if no label was given.
>
> **created**:
> Date when BRU started to create the archive.
>
> **artime**:
> Integer representing the archive creation time. The value of artime is the number of seconds since January 1, 1970 (UNIX time).
>
> **archive_id**:

> A hexadecimal number that can be used as unique archive identifier. This number is derived from the time (first 8 digits) and the process ID (last 4 digits).
>
> **volume**:
> Which volume of the archive this tape belongs to.
>
> **release**:
> The release of the BRU software used to create the archive.
>
> **variant**:
> Information about the particular variant of BRU. A revision control string.
>
> **bufsize**:
> Buffer size (in bytes) used to create this archive.
>
> **msize**:
> Media size (in bytes) in effect when archive was created.
>
> **msize_blks**:
> Same as above, except media size is specified as the number of 2K blocks.
>
> **serial_number**:
> The serial number of the BRU license doing
>
> **device**:
> The name of device used to create this archive.
>
> **user**:
> The user who created this archive.
>
> **group**:
> The primary group of the user who created this archive.
>
> **system**:
> Information about the system on which this archive was created.
>
> **bru**:
> Information about the version of BRU used to create this archive.
>
> **command_line**:
> The command used to create the archive.

Archive Verification

Reporting File Differences: –d

If the –d (differences) option is included on the BRU command line, BRU compares the archive (tape) file with the corresponding file on the disk. If any differences are found, they will be reported and the information will be written to stdout. The amount of information reported is determined by the difference level (the number of –d options) and the verbosity level (number of –v options). **NOTE**: The –d option will read the archive first, and then compare it with the corresponding file on disk. If there is no corresponding file on the disk, BRU issues a warning message. If the disk file differs from the archived file, BRU reports the difference.

If there are disk files that are not on the archive tape, BRU is not aware of the files and no messages will be generated. Differences will be reported only for files that exist on the archive tape.

BRU counts the number of files that are different and lists the count in the execution summary. If there are no differences, the count will be zero. Also, if any differences are found, the exit code returned by BRU will be set to 1.

You can use the –d option when you create an archive:

```
bru -c -d
```

Or you can use it alone, to check which files have changed since the archive was created:

```
bru -d
```

Using the –d option causes BRU to report whether a file has changed either its size or its contents. BRU compares the file and the archive as byte-streams, that is, it treats both files as a stream of characters and reports any difference between the two.

Like –v, –d allows you to specify levels of difference checking. If you call BRU with the -dd option, in addition to comparing the files and the archive as byte streams, BRU reports the following:

- Differences in the file modification date (mtime)
- Changes in the access mode (chmod has been used on the file)
- Changes in the number of links for non-directory files

- Differences in the contents of symbolic links
- Differences in the owner identification (chown has been used on the file
- Differences in group ID (chgrp has been used on the file)

When you specify the option as –ddd, BRU also reports

- Differences in host device
- Differences in major/minor device for special files
- Time of last access for regular files

If the option is specified as –dddd, BRU reports all other differences except for the time of the last status change, major/minor device numbers for non-special files, and size differences for directory files (there may be empty directories).

The –dddd mode is generally meaningful only during a verification pass of a full backup on a filesystem that no users are currently accessing (that is, no files are being created or modified).

Difference Mode Examples

The following example illustrates how to check for files that have changed their size or contents. In this case, we want to list only the names of the files that are different:

```
$ bru -d
"./file1": size contents
```

The next example is similar, except that single level verbosity is specified. In this case, all the files will be listed and each difference will be reported on a separate line:

```
$ bru -d -v
./file1
"./file1": file size different
"./file1": file contents different
./file2
./file3
```

Now we'll increase the difference checking level to detect any changes in the file modification date or in file permissions. No verbosity is specified, so only the differences will be reported:

```
$ bru -dd
"./file1": size mtime contents
```

Any combination of –d and –v options can be specified. You can also use the –V (capital V) option to generate an execution summary that shows the number of archive files that are different. Here's an example:

```
$ bru -d -V
**** bru: execution summary ****
Started: Tue Mar 14 08:50:19 1995
Completed: Tue Mar 14 08:50:53 1995
Archive id: 2f65ba6b0193
Messages: 0 warnings, 0 errors
Archive I/O: 0 blocks (0Kb) written
Archive I/O: 110 blocks (220Kb) read
Files written 0 files (0 regular, 0 other)
Files read 66 files (66 regular, 0 other)
Files in headers: 60
Write errors: 0 soft, 0 hard
Read errors: 0 soft, 0 hard
Checksum errors: 0
Difference count: 1
```

In the event that differences were encountered, the different file will be listed in BRU's output and the "Difference count" field would indicate the number of differences:

```
$ bru -ddddvvvvvf ntape0
...
"/Applications/BRU UB/bruTalk.app/Contents/Resources/
talk48x48Mask.gif": inode number different
"/Applications/BRU UB/bruTalk.app/Contents/Resources/
talk48x48Mask.gif": time of last modification different
d 68788K [1] -rw-r--r--  1 tjones    admin           0
Jul 15 2010 1129350 /Applications/BRU UB/Icon
bru: [I181] read 34616 blocks (69232 KBytes) on volume
[1], 0:00:00, 99999 KB/sec

**** bru: execution summary ****

Started:          Sun Mar  6 09:47:21 2011
Completed:        Sun Mar  6 09:47:21 2011
Archive id:       4cb86b367287
Messages:         0 warnings,  0 errors
Archive I/O:      0 blocks (0KB) written
Archive I/O:      34626 blocks (69252KB) read
Files written:    0 files (0 regular, 0 other)
Files read:       243 files (220 regular, 23 other)
Files skipped:    0 files
Volumes used:     0
Write errors:     0 soft,  0 hard
Read errors:      0 soft,  0 hard
Checksum errors:  0
Difference count: 242
```

Using BRU To Find System Problems

Many UNIX system problems are caused by files that have changed (or are missing). BRU's differences mode can often help you in trouble-shooting these types of problems-if you do a little advance preparation. First, use BRU to create a "reference" backup of your entire system (or just the basic UNIX system programs in the root filesystem). This should be done when your system is properly configured and working properly. Save this reference archive for the future (make sure that the tape is write-protected and properly marked with a label).

If you experience system problems later on, you can use the reference archive with the −d mode to detect any changes in your system. Quite often, this will help you in tracking down difficult problems. The reference archive tape might also be useful in case you need to recover from a system crash.

AUTOSCAN Verification

A large percentage of backup problems are caused by tapes (or other media) that are unreadable. If your tape drive goes bad, it can write tapes that contain errors. Or your tape may contain bad spots. Tape drives are notoriously poor at detecting errors while writing. Usually no errors will be reported and problems will not be detected unless the tape is actually read.

Unfortunately, this is often too late. You may be trying to restore the only copy of a file from a backup tape that was never verified. Unless your archive tapes are verified immediately after you create the archive, any tape drive (or bad tape cartridge) problems may go undetected. Without verification, your backup tape may be worthless.

The AUTOSCAN feature automatically detects problems of this type. AUTOSCAN is enabled by default. With AUTOSCAN enabled, BRU will automatically rewind your tape and "scan" it by reading all the data that was just written. It will read each block of data and verify the checksums. If there are no problems, BRU will continue with the backup and ask for the next volume (if any). If errors are detected, BRU will issue warning messages.

The AUTOSCAN feature is normally enabled when the −c (create mode) option is used.

When AUTOSCAN Is Disabled

AUTOSCAN is disabled for devices that have the norewind field specified in the brutab file (since BRU cannot rewind the device, it won't be able to read it). It is also disabled when writing to devices with unknown parameters (devices that are not listed in the brutab file).

AUTOSCAN is automatically disabled if the −i or −d options are specified – that is, if BRU is invoked in inspect mode or in differences mode. (See Appendix C, "The BRU Manual Page," or Appendix D, "Table of BRU Modes and Options," for a description of all of BRU's modes.) Since these options also read the tape and detect any errors in the data, AUTOSCAN would be redundant. You can disable AUTOSCAN by specifying the noautoscan field as part of the brutab entry for a device. This is not recommended, however. For most tape drives, this scan of your archive tape will take only a few extra minutes (it is typically much faster than writing) and it can help you avoid many future problems.

<div align="right">

17
Command Line Extracting Files: The
Restore Function

</div>

Overview

This chapter describes the command line options used to get files out of an archive—BRU's restore mode. The –x, or extract, option allows you to do a full restore or a partial restore. You may want to do a full restore, for example, when a hard disk has crashed and you have replaced it with an empty formatted disk. You may want to do a partial restore to install a software update, or more commonly, when a user has deleted a file accidentally.

This chapter follows the same outline as Chapter 5, "Archiving Files: BRU's Backup Function." Since many of the command line options are used with both backup (–c) and restore (–x) modes, they will be summarized in this chapter—but the examples will be exclusively of their use with the extract option. You may want to look at the corresponding sections in Chapter 5, or at the BRU manual page in Appendix C, for a more complete description of these options. We've organized the manual this way for two reasons: First, separating the backup and restore functions is more useful for reference. Although backup and restore command lines may include some of the same options, their use is different. Second, by the time you are restoring files, you will be familiar with many of the command line options common to both backup and restore functions. You will not need to hear about them again in detail.

The options that can be used with the bru command allow you to tell BRU:

- That you want to restore files
- What device to use
- The verbosity level
- To generate log files
- To wait for confirmation of each action it takes
- To change the ownership of extracted files
- To specify files using wildcard name expansion
- To extract files regardless of dates
- To extract only files that do not already exist on the system
- To extract files from an archive using a list
- To extract files from an archive based on modification date
- To extract files from an archive based on ownership
- Where to put the extracted files

NOTE: In this section we talk about extracting files from archives. If you are unfamiliar with this way of referring to the restore function, see the "Definitions" section of Chapter 1.

The BRU Command Line

This is the BRU command line format:

```
bru modes [options] [files]
```

bru is the command itself and modes tells BRU, for example, whether you are backing up or restoring files. The most commonly used options specify the device and the level of verbosity. The files tell BRU which files or directories to extract. The sections described in this chapter show how each of the most common options is used.

Command Line Options: Modes

The mode tells BRU the action to perform. For example, the –x mode extracts files from an archive, the –t mode lists the contents of an archive, and the –c mode creates an archive. There must be at least one mode option on the BRU command line. This chapter describes BRU in extract (restore) mode.

Telling BRU To Restore: –x

STOP! Before you extract files from an archive, write-protect your tape (or other media). This will avoid data loss that can result from entering the wrong command. It's easy to make a mistake, like entering bru –c instead of bru –x. The –x mode tells BRU that you want to extract files from an archive. It does not take a parameter. All uses of bru to extract files must begin with

```
bru -x
```

The command line shown above is the simplest extract command you can execute. It assumes two things: First, since it does not specify a device, it assumes that your /etc/brutab file has been set up to specify a default device. (See "Telling BRU What Device To Use: –f," below; Chapter 4, "Setting Device Parameters: The brutab File "; and Appendix E, "A Sample brutab File.") Second, it assumes you want to use BRU's default value for files. When reading an archive, the default for BRU is to select all files in the archive. **NOTE**: By default, BRU will **NOT** overwrite files unless the file on the archive is newer than the file on disk (refer to the –u option).

Command Line Options: Control Options

Telling BRU What Device to Use: –f device

The -f option is used to specify the archive device. If the –f option is not used, BRU will use the first device listed in the brutab file. The example below will create an archive from the files in the /usr/src directory, using a tape inserted into the drive named tape0. The –f option requires a parameter.

```
cd /usr/src
bru -c -f tape0
```

Setting the Verbosity Level: –v

The –v option tells BRU to print information about the procedure it is performing. The –v option provides five levels of verbosity and a summary mode: –v, –vv, –vvv, –vvvv, –vvvvv, and –V (capital V), where a single v represents the lowest and five v's the highest verbosity level. –V (capital V) produces an execution summary. The following is an example of verbosity output:

```
$ cd /
$ bru -xvvvv -f ntape0
x   2k of   4k [1] ./
x   2k of   6k [1] ./bin
x  72k of  78k [1] ./bin/acctcom
x  68k of 146k [1] ./bin/adb
x  36k of 182k [1] ./bin/ar
x 120k of 302k [1] ./bin/as
  . . .
$
```

The first field in the output contains the mode character, as a reminder of what mode is currently running (in this case the "x," or "extract" mode). The next field gives the amount of space in kilobytes this particular file will use in the archive. The next field ("of XXXk") gives a running total of the amount of data in the archive, including the current file. Note that the first value is 4k. This represents the 2K header block for directory "/" and 2K for the archive header block. The number inside the square brackets is the current volume number. Finally, the name of the current file is listed.

Generating Log Files: −v

If you want to keep a record of how a particular tape archive was extracted, you can redirect the output of any of the verbosity options described above to a log file. For example:

```
bru -xvf tape0 > extract.log
```

This will write a list of the extracted files to a file named extract.log.

Label Option −L

The −L option during restores is used as a comparison. BRU will read the label string given and attempt to match it to the label of the tape or archive device. If the label does not match on the first tape, BRU will abort the operation. If the label does not match on subsequent tapes, a warning is issued, but the extraction or restore will continue.

There are two ways of providing this label to BRU. The first is to simply specify the label from the command line as shown below:

```
bru -xv -L "Test Label" /usr/bin
```

The second method of providing the label string to BRU, is to create a file and tell BRU to read that file. In this fashion BRU will read in the first 63 characters of the file as the label for the tape as shown below:

```
bru -x -L "/tmp/label name" /usr/bin
```

The label information appears in the archive header, which can be viewed (or fed to an archive manager program) using BRU's −g option. **NOTE**: That the label string and file name are enclosed in a pair of double quotes. This assures that any shell program you might be running (sh, csh, ksh, tsh) will treat the label as a single parameter. If the label is not enclosed in double quotes, BRU will see the individual components of the label string as separate parameters. In the second case, /tmp/label would become the label, and each of the words in the rest of the label - in this case 'name' - would be treated as a file name.

Command line override option: −Q

The −Q options along with the −x "Restore" option change default BRU operations as described below. Below we have laid the options out in the command line format −QL for instance.

-QL

Use a literal string as a tape label. This will override BRU's attempt to look for a file from which to read the tape label. This can be useful if you have a file that is the same name as the label you wish to apply to the restore command.

-QR

Disable SmartRestore. This option turns off BRU's handling of open or shared files. It is not recommended that you override this setting

-QS

Translate Symbolic Links. This option is used in conjunction with the −T option will force the translation of symbolic links as explained in the −T option found above.

-QV

Ignore "Incorrect Volume" warnings. When restoring from a multi-tape set or beginning a restore from other then the first tape is a set, BRU will normally issue a warning message. Using this option BRU will not issue the warning message and continue the restore process.

Depth Selection on Restore Option: −U

This option allows you to set the depth at which BRU will work during its restore process. Files more then the -U(#) levels will not be processed. For example, if the current directory is /home/webmaster, −U0 (zero) will only restore the files and directory nodes in the /home/webmaster directory. While the directory node (name) will be restored the files in side each directory will not be restored.

With a level of −U2 (two), BRU will restore /home/webmaster/test, /home/webmaster/test/runone, /home/webmaster/test/runtwo, but not restore /home/webmaster/test/runone/demo1. In this configuration BRU can be used to easily restore a subdirectory from a total system backup.

Confirmation Option: −w

When the −w option is specified, BRU will wait before attempting to extract a file. For each file, it will do the following:

Print the file name, show the action it is about to take, ask for confirmation. BRU also provides a special response that lets you change your mind. If you type a g (for "go ahead") instead of y or n, BRU will stop asking for confirmation. The following example shows how this response is used:

```
$ bru -xvw -f tape0
x ./: please confirm [y/n/g] y
x   2k of   4k [1] ./
x ./bin: please confirm [y/n/g] y
x   2k of   6k [1] ./bin
x ./bin/acctcom: please confirm [y/n/g] y
x  72k of  78k [1] ./bin/acctcom
x ./bin/adb: please confirm [y/n/g] y
x  68k of 146k [1] ./bin/adb
x ./bin/ar: please confirm [y/n/g] g
x  36k of 182k [1] ./bin/ar
x 120k of 302k [1] ./bin/as
x  16k of 318k [1] ./bin/asa
x   4k of 322k [1] ./bin/basename
. . .
```

Changing Ownership of Extracted Files: −C

BRU stores the user and group ownership with the archived files and normally restores files with the original user and group ownership. In some cases it may be necessary to change the ownership of restored files to that of the user and group running BRU. The −C option is not an option you will normally use. It is typically used for extracting files from tapes that were written on a different system (with different user and group IDs). The ownership of such extracted files can be changed by using the −C option (be sure you use a capital C). For a full description, see Chapter 9, "Advanced Uses," "Ownership of Extracted Files."

Command Line Options: File Selection

There are several ways to specify which files you want to extract. For example, you can:

- Use wildcard file name expansion
- Flag files to be extracted unconditionally-that is, without regard to date
- Tell BRU not to overwrite any existing files
- Extract files read from the standard input
- Extract files by date
- Extract files by user
- Use include/exclude patterns
- Tell BRU where to put extracted files: Relative and Absolute pathnames

Be careful to specify file names last. If your file specifications are not in the correct position on the command line, the results of the command may be unexpected. If files are not specified, BRU defaults to all files on the archive tape. That is, the command

    bru -x

is equivalent to

    bru -x "*"

Using Wildcard File Name Expansion

BRU recognizes wildcard strings in the same format as recognized by the Bourne or Korn shells. However, when such strings are passed to BRU in extract mode, they must be quoted to prevent their expansion by the shell. For example, the following command will extract all files in any subdirectory one level below the current directory that start with any character between "a" and "h" and end with ".c"; it will also extract all files in any subdirectory three levels down:

    bru -x "./*/[a-h]*.c" "./*/*/*"

The command in the next example will extract all files in the ./etc subdirectory that do not end in ".old" (the "!" operator is a BRU enhancement of wildcard expansion that does not exist in the shell):

    bru -x "./etc/!*.old"

Use the "!" operator with extreme caution. In particular, since each pattern is applied independently to determine a match, be wary of including more than one pattern with a "!" operator on the command line. For example, the pattern "!file1 !file2" will match all files, including files "file1" and "file2". Since "!file1" matches pattern "file2" and "!file2" matches pattern "file1", you have defeated the exclusion.

The "!" is mostly useful for simple exclusion tasks such as excluding a single file. For other file exclusions, it is usually easier to use the -X option. See Chapter 6, "File Inclusion and Exclusion." **NOTE**: Wildcard characters used to specify file names in -x mode must be enclosed in double quotes. Wildcard characters used with -c are not quoted.

Using Include/Exclude Patterns

The -QX option can be used to select files when restoring. Refer to Chapter 6, "File Inclusion and Exclusion," for details.

Unconditional File Type Extraction: -u flag

Normally BRU will not supersede existing files while extracting files from an archive - that is, BRU will not overwrite an existing file with an older archived file of the same name. Specifying the -u flag option causes files of the type specified by the flags listed below to be unconditionally selected for extraction. Such files will overwrite any existing files of the same name regardless of modification times. If the verbosity level is two or higher (-vv through -vvvv), BRU will print a message for each file that is not superseded. The confirmation option (-w) may also be used with the -u flag option.

The available flags are:

      a   Select all files
      b   Select block special files
      c   Select character special files
      d   Select directories
      l   Select symbolic links
      p   Select fifos (named pipes)
      r   Select regular files
      f   Select regular files (the same as r)

Multiple flags select all files that match any of the given types. For example, –ubcd will select all block, character, and directory files. The –ua option will select all files and is equivalent to specifying each of the other individual flags.

Existing directories are never overwritten. If you select directories, only their attributes may be modified. The –u option merely allows directory attributes to be set back to some previously existing state.

If you select symbolic links, only the contents of the link will be modified. It is impossible under some versions of UNIX to change the access time, the modification time, or the file mode of a symbolic link.

The following example will extract all regular files in an archive, regardless of their modification dates. Any existing files with the same names will be overwritten. If any of the extracted files has been changed since the archive was created, all such updates will be lost.

```
bru -x -ur
```

The next example specifies that all of the files under directory ./usr/doe are to be extracted from the archive on the default device, and are to unconditionally overwrite any existing files with the same names:

```
bru -x -ur ./usr/doe
```

The –E File Extraction Option

When extracting files from an archive, BRU will normally replace an existing file only if the corresponding archive file has a more recent date. For example, if the existing file has a modification date of January 1, 1990, it would be overwritten by an archive file (of the same name) with a date of February 1, 1990. Most of the time, this is the desired behavior.

In some cases, you may not want to overwrite an existing file, even if the archive file is newer. This is often true when doing software updates. For example, a software update may contain configuration files with dates that are newer than existing configuration files. If these carefully created configuration files are overwritten, the software may not work properly and the configuration files will need to be re-created.

The –E option lets you avoid these problems. If –E is specified on the extract command line, BRU will not replace any existing file, even if the archived file is newer. It will only extract files that do not currently exist. The following is a typical use of the –E option:

```
bru -xv -E
```

Of course, if you really do want to extract everything from an archive, use the –ua option. This will perform an unconditional extraction. BRU will overwrite all files with the same names as the archived files-regardless of the date. See "Unconditional File Type Extraction: –u," above. **NOTE**: The –E and –ua options should not be used together.

Extracting Filenames Read from stdin: –

The hyphen (–) is a special filename. When it is specified on the BRU command line in place of the files option, it tells BRU to read the standard input stream (stdin) for a list of files to process rather than collecting them from the command line. The rules governing the use of the hyphen as a special filename are the same for both reading and writing archives. This usage takes the form:

```
bru -x [options] -
```
This option is typically used in conjunction with a UNIX pipe or with the standard input stream redirected to a file containing a list of filenames. The commands

```
cat /tmp/filelist | bru -x -
```

and

```
bru -x - </tmp/filelist
```

are equivalent and extract the specific list of files you have named. **NOTE**: filelist must contain all the files, that you wish to restore (not just the directory names). filelist must also be in the same order. You may wish to list the files with 'bru –tv >filelist' then edit filelist for the files

you want to restore. **NOTE**: BRU will read your list of files BUT WILL NOT DO WILDCARD EXPANSION of that list of files. BRU expects the list of files to be completely expanded when it receives the list. Therefore WILDCARDS CAN'T BE USED in your list of files for restoring data. **NOTE**: If you use a hyphen (–) in place of a list of files on the BRU command line, you are not allowed to use the "–f –" option to read an archive from the standard input on the same command line. These two usages conflict. The following command is illegal:

```
bru -x -f - - <filelist
```

It is not possible for BRU to read both the archive and the list of files from the standard input.

Extracting Files by Date: –n date

The way dates are specified is the same for both backup and restore functions. The command line differs only in the use of -c for backup and –x for restore. In the case of a backup, you are asking BRU to archive files created or modified after the date you specify on the command line (see Chapter 5, "Archiving Files: The BRU Backup Function," "Selecting Files by Date: –n"). In the case of a restore, you are asking BRU to extract files from an existing archive based on the dates on which the files were last modified.

For the details of date specification, see Chapter 14.

Inverse Date Specification

If you need to extract files older than a given date, you can do this by placing an exclamation point (!) immediately before the date specification on the command line. The exclamation point reverses the sense of the match. For example,

```
bru -x -n !14-Mar-92
```

will extract all files created or modified before March 14, 1992.

Extracting Files by User: –o

File extraction can be limited to files owned by a particular user in the same way files are selected by user when you are creating an archive. The –o option lets you tell BRU the identity of the user whose files you want to extract from an archive. BRU accepts the following three forms of user identification:

- User name
- Numeric user ID
- File owner

Although these three ways of using the –o option were discussed in Chapter 5, "Archive Creation: The BRU Backup Function," examples of their use in restoring files may be useful here. If you have questions, see Chapter 5, which goes into greater detail.

Extracting Files by User Name: –o username

The user name must correspond to a user in one of the password files. It identifies the owner of the archived files you want to extract. This will extract all files owned by the user bob:

```
bru -xv -o bob
```

Extracting Files by User ID: –o uid

You can also specify the owner by the decimal identifier derived from the uid value in the /etc/passwd file. The decimal value is used less often than the other forms of the -o option. This command line will extract all files owned by the user with user ID 112:

```
bru -xv -o 112
```

In some circumstances, it is possible for files to have user ID numbers that do not match any user listed in the /etc/passwd file.

Extracting Files by File Owner: –o pathname

If you use the pathname to a file as the argument to the –o option, BRU assumes that the owner of that file is the user whose files are to be restored. BRU selects all the files that are owned by the owner of pathname. For example, if owner.file is owned by bob, BRU selects all files owned by user bob:

```
bru -xv -o owner.file
```

Where To Put the Files: Pathnames

The placement of extracted files is largely determined by the way in which files and directories were specified when the archive was created. Archived files may have been stored using either relative pathnames (pathnames that begin with a "./" or an alphanumeric character) or

absolute pathnames (pathnames that begin with a "/" character). An absolute pathname always begins at the root of the system.

If the archive was created by specifying an absolute pathname, for example, `/usr/me/myfile`, then the files will be restored in that exact location.

If the file specification at the time of archive creation, did not begin with the "/" character, for example:

> **./currentstuff/myfile**

or

> **currentstuff/myfile**

then the files will be restored in the current directory when BRU is run to extract them. In the above case, if `/tmp` is the current directory, then the file will be restored as

> **/tmp/currentstuff/myfile**

If you specify a relative path when you back files up, then the BRU archive tape can be used to restore a filesystem to a different device than the device used to create it. An entire directory can be moved from one machine to another without first determining that there is enough space on a particular filesystem to install it. You simply have to choose a filesystem where there is enough space and issue the command to extract the directory from a location in that filesystem. The relative pathnames of the files assure that the installation occurs in the "current directory."

There are no particular advantages to specifying absolute pathnames when creating a BRU archive, and in general, absolute pathnames should be avoided.

NOTE: When trying to match filenames,

> **abc/xyz**

is **NOT** equivalent to

> **./abc/xyz**

The filename must be specified in exactly the same way as the name of the file stored in the archive. You can use

> **bru -tv**

to see how the name is stored. See Appendix C, "The BRU Manual Page," for a description of the –t option.

Translate on Restore Option: –T file

Translate on restore allows the user to translate, rename or relocate files based on the content of a translation file. The file should be created in a two-column format. The first column should list the files you want to have translated, the second column should list the new directory or name you want the files translated to. This translation applies to directories, names and extensions. By default, symbolic links will not be translated. See the –Q option mentioned earlier in this manual. To help in the understanding of this option we will work with this example. To restore all of the files from /home/bill and place them in `/home/paul`, you would create a file that had the following text. For this example we will call this file trans.

Contents of the file trans

> **/home/bill /home/paul**

Command line:

> **bru -xvvv -T /dir/transfile /home/bill**

No further interaction is then needed. During the restore BRU will translate all files in the `/home/bill` directory to the new `/home/paul` directory. You can also have bru handle multiple translation. Simply by adding more lines to the same file you could have bru translate all of the files from many directories to different directories. See example below:

For Unix and macOS:

> **/home/bill   /home/paul**
> **/usr/lib   /usr2/lib**
> **/home   /u2**

For Windows you may use either the Windows \ by escaping it (\\) or the more Unix-like /. It's bets to use one style or the other for clarity:

> **/C:\\Users\\argest\\Developer\\. /F:\\Migrate\\Dev**

or

> **/C:/Users/argest/developer/ /F:/Migrate/Dev/**

The translate file can contain as many translation lines as necessary, but each line must consist of a pair of entries. **NOTE**: When using BRU with the Translate on Restore option and the –PA option, BRU will translate those files inside of the translate file and convert the remainder of the files with the –PA option as described below.

Converting Pathnames from Absolute to Relative Form: –PA

If you specified an absolute pathname for the files you backed up (an absolute pathname is one that starts with a slash character "/"), then BRU will attempt to restore them in the root directory with exactly the same pathname. This is fine, unless you want to move the files to a different directory.

If you want to restore files with absolute form pathnames in a different directory, you must use the –PA option. The –PA option will translate the leading slash character from an absolute pathname to "**./**" and permit extraction into the current directory.

Here's an example of how the –PA option is used. First we back up a file with an absolute format pathname:

```
bru -cvf tape0 /etc/termcap
```

Now we restore the file to a different location:

```
cd /u/mydir
bru -xv -PA -f tape0
```

This creates the file `/u/mydir/etc/termcap`. If the –PA option had not been used, the file would have been restored as `/etc/termcap`.

The –PA option is needed only if you created an archive with absolute pathnames. It has no effect if the archive already contains pathnames in relative format (BRU's default mode). **NOTE**: To extract individual files using the –PA option, you must specify the filename without a leading "/" character. For example, if a file was archived as `/usr/bin/man`, the normal command to extract a single file,

```
bru -x /usr/bin/man
```

will restore the file to that exact location. The command you must use to restore the file to the current directory at the time of extraction is:

```
bru -x -PA usr/bin/man
```

Note the difference in the files argument.

<div align="right">

_**18**_
**Command Line Advanced Uses**

</div>

Overview

The most important uses of BRU-creating archives, inspecting or verifying archives, and extracting files from archives have been covered in earlier chapters. In this chapter, we describe some of the more advanced uses of BRU. Among these advanced uses are:

- Using a different I/O buffer size
- Using an explicit media size
- Reading archives from stdin or writing archives to stdout
- Using multiple archive devices
- Controlling ownership of extracted files
- Ways to increase BRU's speed
- Saving archive space by using data compression
- Setting up BRU to run with brutalk
- Setting the interaction options
- Running BRU from cron
- Controlling BRU's I/O streams
- Tuning BRU for shared memory
- Using BRU to save and restore sparse files
- Using BRU with a bootable floppy
- Using BRU with remote backup devices
- Using BRU to do "live" system backups

Telling BRU the Buffer Size: -b bufsize

You can tell BRU to use a specific buffer size for reading and writing data. Normally BRU uses a default buffer size if no buffer size for the output device is specified in the brutab file. However, you can change the value BRU uses by specifying the -b option on the command line. This option specifies the size of the data block that is written or read with each access to the archive device. When writing, BRU stores the buffer size information in the archive header. When reading the media, BRU will use this buffer size (unless a different value is specified by the -b option).

When you use the -b option, you tell BRU to use a buffer that is bufsize large. As with other numeric parameters, you can specify the size in bytes, kilobytes, or megabytes. A typical value is 32K, used as the default

value. When you provide a parameter to specify the buffer size, that parameter must be an even multiple of 2K (that is, a multiple of 2048 bytes). A value of 2K is the absolute minimum size that BRU will accept. The maximum value is determined by your hardware and system software. If you specify a value that is not a multiple of 2K, BRU will round up to the next direct multiple of 2K. Here is an example that specifies a buffer size of 64K:

```
bru -b 64K [options] [files]
```

NOTE: Some tape devices have fixed buffer sizes. If you have such a device, you must set the BRU bufsize to a multiple of the device's buffer size or your write attempts will fail with a write error. If you experience errors, try adjusting the buffer size and writing small test backups until the write succeeds. This value, or some multiple of this value, will be the best setting for the device.

Setting the Archive Media Size: -s msize

You can use the -s option to tell BRU the capacity of the media you are using. With most modern operating systems and tape devices, this setting is not necessary. However, you may wish to set the media size for devices like removable disks or disk files. **NOTE**: When you use this option, it overrides any value you may have specified in the brutab file with the size= parameter. If you have two or three different sizes of tape that you use for your backups, but only one tape drive, you can use the -s option each time you run BRU to specify the size of the tape you'll be using. However, this is tedious, and specifying a size larger than the actual size is a serious error.

A better solution is to create "aliases" for the tape drive, and change your brutab file appropriately. Simply duplicate the basic brutab entry for the device, but change the device name and size= parameter. Here's an example:

```
/dev/tinytape size=10M [other parameters]
/dev/medium   size=20M [other parameters]
/dev/bigtape  size=30M [other parameters]
```

Now create links for each new device name:

```
$ cd /dev
$ ln st0 tinytape
$ ln st0 medium
$ ln st0 bigtape
```

The paths /dev/st0, /dev/tinytape, /dev/medium, and /dev/bigtape all point to the same device, but BRU considers them to be separate devices with the characteristics given in the brutab file.

When the –s option is specified, it overrides any other default value, even that read from the tape header during a read or scan of the archive. The value that you give with the -s option should be an integral multiple of the BRU buffer size. However, BRU does not indicate any error when the value is not a multiple. Instead, BRU calculates the effective media size by silently rounding down to the nearest multiple of the buffer size.

One very handy use of the –s option is to create an archive in a set of normal files, for transfer via modem or email to another system (using mail, ftp, or another file transfer utility) where the receiving system limits the size of incoming attachments. By splitting the archive into several smaller pieces, large archives can be transmitted in small chunks, reducing retransmission overhead in case of errors. For example:

```
bru -cv -s600K -f bru1 -f bru2 -f bru3 [files]
```

will cause BRU to write its archive into the files called bru1, bru2, and bru3. Each file would have a maximum size of 600K bytes. These files could then be sent to another system (remote) with the UNIX scp (secure copy) command:

```
$ scp bru1 user@remote:/usr/spool/uploads
$ scp bru2 user@remote:/usr/spool/uploads
$ scp bru3 user@remote:/usr/spool/uploads
```

and unpacked on that system with the commands:
```
$ cd /usr/spool/uploads
$ bru -xv -f bru1 -f bru2 -f bru3
```

Reading stdin or Writing stdout (standard I/O streams)

If the pathname supplied as the parameter for the –f option is a hyphen (–), then BRU uses the standard input (stdin) for archive reading or standard output (stdout) for archive writing, as appropriate. When extracting files from an archive, (–x option), the "–f –" causes the input archive to be read from stdin. The filenames to be extracted must be in the same order as they appear in the archive (or files will be skipped). If the filenames are not in the same order, you must use the –Pf option.

This will cause BRU to sort the names (by building an internal filename table) before it does the extraction.

An example would be extracting a gzip'd BRU file in a single command line:

```
gunzip -c gzipfile.bru.gz | bru -xvf -
```

This allows gunzip to expand the compressed file and pass the results via stdout/stdin through to BRU for extraction.

To perform the reverse operation, you can instruct BRU to send its output to gzip for compression as follows:

```
bru -cvf - /files | gzip > brufile.bru.gz
```

The resulting file will be a BRU archive that has been compressed with gzip. This can also be accomplished using BRU's external compressor option, described later in this chapter.

Using Multiple Files/Devices: –f device

You may use the –f option multiple times on the same command line to tell BRU to use multiple files or multiple paths (which could be devices, for example) to store the archives. If multiple –f options are given, each path is added to a list of paths to cycle through each time a volume change is required. When the end of the list is reached, BRU automatically cycles back to the first path and waits for confirmation to continue the cycle again. Any input other than a carriage return will cause BRU to use the newly entered path and to abort the cycling for the remainder of the current run. This feature is known as device cycling.

The following example will do a full system backup (starting at root). It will first write data to tape1. When it is full, it will automatically continue on tape2. **NOTE**: When using this option all devices need to be the same size as the first device. BRU takes the size of the first device and uses that size for all other device when device cycling starts. If you specify a size of zero (0), BRU will then continue to write to each device until it receives an end of tape message. Then, BRU will switch to the new drive or device:

```
bru -cv -f tape1 -f tape2
```

To do a multiple device restore, the command is similar:

```
bru -xv -f tape1 -f tape2
```

BRU commands run from cron often use the device cycling feature. This allows BRU to get as much work done as possible before interaction is necessary. Device cycling can be used with the global brutab parameters MOUNTCMD and UNMOUNTCMD to allow writing to jukeboxes or tape libraries. If these parameters are used, then BRU will continually cycle through all the devices until it has completed reading or writing. It will not wait between devices or when it returns to the first device-any waits should be done by the shell scripts specified by MOUNTCMD and UNMOUNTCMD see Appendix K, "Using MOUNTCMD and UNMOUNTCMD".

Ownership of Extracted Files: –C

When extracting files, BRU restores the owner and the group to that which is stored in the archive for each file. By specifying the –C option, you instruct BRU to change the owner and group ID to that of the person running BRU, including that of root. If the system administrator root is performing a filesystem restore, this option should not be used, since root would end up owning all of the files.

The –C option is useful when you are importing files and directories from other systems because it allows the person running BRU to assign all of the files to himself. It is best to check the archive by running BRU with the –tv options to see who owns the files in the archive and thus to determine whether or not the –C option is appropriate to use.

BRU enforces standard UNIX access security. It makes every attempt to prevent normal users from archiving or extracting files to which they might not normally have access. If a user does not have read access to a file, that user will not be able to back it up, restore it from an archive, or change its ownership with the -C option. **SECURITY NOTE**: Anyone with physical access to the archive media and a knowledge of the structure of the file records can recover the archive's contents by writing their own file extraction program. If data security is important, you should protect the archive media from unauthorized physical access.

Increasing BRU's Speed

BRU has a number of options that can, at times, increase the operating speed of the program. These options include using a larger I/O buffer and using double buffering.

Before attempting to tune BRU for optimum performance, you should know the maximum speed of your archive device. On many newer

systems, BRU is limited by the speed of the tape drive. If your tape drive is already operating at top speed, it is impossible to adjust BRU's parameters to make it run faster.

Listed below are maximum native speeds for various types of tape drives:

Drive Type	Speed
DDS4	2.5MB/sec
DAT160	6MB/sec
DAT320	12MB/sec
LTO-1	15MB/sec
LTO-2	30MB/sec
LTO-3	60/80MB/sec
LTO-4	80/120MB/sec
LTO-5	150MB/sec

The above speeds are the native throughput for the drives listed. Hardware data compression will increase these values depending upon how well the data compresses. The numbers are based on information gathered from the tape drive manufacturers. This list is meant only as a guide, the actual maximum speed of your tape drive may be different. To obtain the rated maximum speed for your drive, refer to the manufacturer's specifications.

During the I/O operations, BRU calculates the tape read/write speed and records the value in its execution log, /var/log/bruexeclog. Here is a typical log entry (this would be on a single line in the log):

```
20010903 04:19:10|4331|root|[L182] wrote 256670 blocks
on volume [1], 0:48:59, 174 Kb/sec
```

In this case, BRU was writing to a 4mm DDS1 drive. It took 2,939 seconds (0:48:59) to write 513340 Kbytes (256670 x 2 Kb/block), so the tape speed is 174 Kb/sec. This is very close to the maximum DDS1 speed of 180 Kb/sec, so it is likely BRU is "tuned" for optimum performance. Any further adjustments would probably be futile.

TUNING HINT #1: Make sure you read or write enough data to get an accurate tape speed measurement. Some types of tape drives (like DATs) take several seconds to load the tape before they actually begin reading or writing. This extra time will skew the speed measurement (the apparent speed will be slower) on small amounts of data. To avoid this problem, run your tape speed tests for two minutes or longer. The longer the test, the more accurate the speed measurement will be.

TUNING HINT #2: Do not use BRU's compression (−Z option) when tuning for optimum speed. Data compression can be used after you have determined the optimum parameters. If your tape drive supports data compression (hardware compression), it is usually faster than BRU's compression. If this is the case, BRU's compression option should not be used.

Specifying a Larger (or Smaller) Buffer Size

BRU operation can sometimes be performed more quickly if a larger buffer is specified, particularly with streaming tape drives. However, on some systems, a smaller buffer size will result in greater speed. The default buffer size, if the −b option is not specified, is read from the brutab file. Normally, this is set to bufsize=32K. If you use a larger buffer, two things happen:

1: Your performance might improve due to larger blocks being read and written in a single action to the archive device.
2: If you are using a tape unit as the archive device, using a larger buffer may allow you to store more data on the tape since the proportion of tape that stores data goes up as the number of inter-record gaps decreases.

A smaller buffer size (like 2K or 4K) is often faster when using tape devices that have small buffers or on systems with tape controllers or device drivers that do their own buffering.

There are several things that can affect the size of the buffer and therefore the tape block size that is written to the archive:

Available memory size

If you don't have enough free memory to allow this size buffer to be used, BRU will issue an error message and quit.

Device limitations

If your archive device driver has a specified limit on the block size that it can support and you have specified a buffer that is larger than that size, then you may experience problems. Even though the amount of memory will be allocated for BRU to use, reads or writes to the archive device will fail, usually with the error message "no such device or address".

Improperly specified size

If you specify a size that is not an even multiple of 2048 bytes (also called 2K), the buffer size will be silently rounded up to the next multiple of 2K.

Archive header information

If you are extracting files, the buffer size read from the archive header takes precedence over the default buffer size or a buffer size read from the brutab file for the given archive device. Any buffer size option given on the command line takes precedence over any other source of the buffer size.

Double Buffering: −D

If your system uses System V shared memory (your system administrator can tell you if it does), then depending on the constraints of the archive device driver, using double buffering (the −D option) can sometimes show a dramatic increase in the I/O rate of the archive device. It is not uncommon to see the speed increase by a factor of two. Unfortunately, it is also not uncommon to see the speed decrease slightly also. This feature is very hardware and software dependent, and the only way to discover if you should use it for a particular device is to try it and see if it helps.

Turning Off Error Checking: −F

DO NOT USE THIS OPTION! If that didn't scare you off ... running BRU with the −F option disables calculation and checking of error-sensing values. You'll generally only use this option when you are absolutely sure that the archive media is essentially perfect; that there is no chance for an error to occur in the recording process.

If you record an archive with the −F option, you must read it back with the same option. **NOTE**: That some of BRU's automatic features, such as AUTOSCAN and byte swapping (mentioned in the Introduction), are not functional if error checking is disabled.

The option is provided primarily to facilitate the output of one BRU (perhaps that creates an archive) to be piped into the input of a second BRU, that extracts data from an archive. In this case, BRU becomes a fancy copy program. For example, to copy all of the files from /usr/u1 to /usr/u2, the following command could be given:

```
$ (cd /usr/u1; bru -cFf -) | (cd /usr/u2; bru -xFf -)
```

Note that in this case, the "–" at the end of each of the two parts of the command follows the use of the -f option. In this case, the "–" does **NOT** mean that BRU should look for a file list to be provided by reading the standard input as explained for the "–" option in an earlier chapter. Instead, the hyphen is the file path (see the –f option explanation). It may look a little confusing, but BRU knows the difference. If you want a file list to be provided from stdin, just make sure the –f option does not appear on the command line with a conflicting "–" usage. This option can also be used to transfer files to a different system. As long as BRU is installed on both machines. In the example below we are using the rsh command. This command may be different on different UNIX platforms.

```
$ bru -cFf - | rsh hostname "(cd /tmp; bru -xFf -)"
```

Using Data Compression: –Z

If you specify the –Z option, BRU will use compression to make the final size of the archive smaller. The compression algorithm supports multiple levels of compression - the higher the level, the more aggressively BRU will work to compress your data. Also, the higher the number, the harder your CPU will be tasked to perform the compression. The default is to use level 3 compression, but you may adjust this for your needs by specifying the –N level option, where level is the desired compression level (1 to 6) Example.

```
$ bru -c -Z -N 6 ./
```

NOTE: The –N option has no effect unless the –Z or –S options are specified. Using this compression technique can result in space savings of 0% to 90% or more depending on the kinds of files being stored. Typically, most files will compress about 30 to 50 percent, however sparse files containing lots of redundant data or zeros, such as typical large database files, may compress as much as 90% or more. If USIZE is the uncompressed size of a file (as shown by ls –l), and CSIZE is the compressed size of a file, BRU defines the compression ratio in percent as:

```
Ratio = 100 * ( 1 - (CSIZE / USIZE) )
```

One experiment, using a "typical" mix of files consisting of some text files, some executable binary files, and some miscellaneous data files, gave the following results:

Typical Compression Ratios:

Level	Archive Size	Ratio
None	7434K	0%
1	4239K	69%
3	4178K	70%
6	4055K	72%

Compression is not the default mode because not all versions of BRU know how to deal with compressed files (early versions of BRU did not support the –Z option).

The disadvantage to the use of compression is that it takes time and extra memory to compress or decompress the files. The overall impact on the archiving time depends on the spare CPU capacity of your system. If BRU is spending most of its time waiting for I/O to complete, then compression may have no noticeable effect on your archiving time. On the other hand, if your system is very busy, or has a slower CPU, then compression can easily double or triple your backup time.

One last hint, the –S option can be used to turn on automatic compression for files larger than a certain threshold size (see the discussion of the –S option later in this chapter). In this case, only those files larger than the specified threshold will be compressed. This is generally much faster than turning on compression for all files, while still providing significant space savings. When using the –S option, it is not necessary to specify the –Z option. **NOTE**: If your tape drive provides hardware data compression, use it instead of BRU's compression option. Hardware data compression is normally much faster and more aggressive than BRU's compression. Since it is generally not possible to gain anything by compressing data that is already compressed, **DO NOT** use both types of compression. If you do, your tape speed will decrease and tape usage will increase.

Using An Alternate Compressor With BRU

BRU 16.0 and later provide a mechanism which allows you to utilize an external compression engine instead of the built in LZO engine. This allows BRU to be used with more modern compression engines as they become available.
To use an external compressor, BRU looks at the global variable "BRUZEXTERN" to determine the program that should be used. You can set this as either an environment variable:

```
BRUZEXTERN="/usr/bin/bzip2" ; export BRUZEXTERN
```

or as a global variable in the `/etc/brutab` file (refer to "Chapter 4 - The BRUTAB File" for more information on global variables):

```
#+BRUZEXTERN="/usr/bin/bzip2"
```

The only requirement is that the compression engine accepts `stdin` as the compression source and send the resulting compressed data to `stdout`. Also, you must be able to call the engine for decompression using a `-d` option to the original program (i.e.: `bzip2 -d`).

Once a tape is created using an external compressor, you do not need to set the BRUZEXTERN for BRU to be able to extract the compressed archive. The setting for BRUZEXTERN is added to the archive header and future uses of BRU to extract files from this archive will know which program to use for decompression automatically.

Using BRU with brutalk: -I args

Using the –I option, BRU can be set to run interactively with a program such as brutalk (see Appendix I, "The brutalk Program"). If there is no terminal present, you can direct BRU to write its queries to a fifo (first-in-first-out buffer file) and read responses from a different fifo. The interactive program, in turn, watches the output from BRU and directs its activities accordingly, responding through BRU's input response fifo, just as though the responses were made from an interactive terminal.

When BRU was first installed on your system. The installation created the two fifos listed below. The commands listed below, can be used to create the fifos if for some reason they where not created during install, or you need to create additional fifos.

```
$ mknod /bru/bru.q p
$ mknod /bru/bru.r p
```

To create different fifos i.e. for running multiple BRU jobs.

```
$ mknod /bru/myfifo.q p
$ mknod /bru/myfifo.r p
```

In order to use the brutalk feature you will need to add the option listed below to your bru command line.

```
-Ib
```

If you created a different set of fifos, (i.e.: if you are running multiple BRU jobs) then you would add the following argument to your BRU command

```
-Iq,/bru/myfifo.q -Ir,/bru/myfifo.r
```

to the desired BRU command line which ultimately gets executed. The first time BRU needs to communicate with an operator, it will open the two fifos, write a query to the `bru.q` fifo, and wait for a response from the `bru.r` fifo. The brutalk program, which is a simple program provided with BRU, can be used to read the query and send a reply. Here is the command to start brutalk if you are running the standard fifos as explained in the previous example:

```
brutalk
```

If however you are writing to a different set of fifos simply run the example below

```
brutalk [-t ttyname] /bru/bru.r
```

or

```
brutalk [-t ttyname] /bru/bru.q /bru/bru.r
```

The second form is used if you want brutalk to timeout after a few seconds if there are no queries ready to be read. The brutalk program will continue to read queries and send replies until either BRU exits, or a CTRL-D (EOF) is typed at the terminal. The optional ttyname parameter can be used to force brutalk to interact with the user via some terminal steam other than /dev/tty. See Chapter 5: "Archiving Files: The BRU Backup Function," "Running without User Intervention: –B." If the –B option is used, BRU is forced to run in the background and will terminate if a problem occurs.

The selection of either method depends on what the desired recovery is when interaction is necessary. If you prefer to have the backup terminate and not rerun, then use the –B option. However, if you want to be able to recover from an unexpected situation (for example, if the tape is simply off line), and to rerun or continue the backup, then use the –I option, and use brutalk to resolve the problem when it is convenient. brutalk can also be used by someone dialing into the system to check on BRU's progress, since it can be run by anyone from any terminal. NOTE: Only the permissions on the read/write fifos enforce security.

Running BRU from cron

Using the interactive option allows you to run BRU from cron (cron runs programs at specified times and frequencies). If no interaction with the user is required, running from cron is no different than running directly from a terminal. However, when interaction is necessary there are basically two options; terminate, or find some way to communicate with the operator (or another program masquerading as the operator). The –B option provides for simple termination while the –I option provides for communication with an operator. BRU recognizes the following parameters for the –I option:

```
b            default fifos
q,fifoname1   write queries to a fifo
r,fifoname2   read responses from a fifo
l,filename    write log information to a file
```

You can use the device cycling feature to specify that more than one archive device contains tapes that are ready for a nightly backup. For example, an operator might load a set of tapes onto several drives, and schedule cron to begin a daily backup at midnight, when all of the programmers and office staff have gone home. BRU will do as much work as it can without interaction, and then wait patiently for an operator to provide additional information in the morning. As an example, assume that the crontab file contains the following crontab entry:

```
30 02 * * * sh -c /etc/dailybru /usr
```

and the file /etc/dailybru contains:

```
# Backup the specified files.
# Use fifos to communicate if necessary.
FILES=$*
IOPS='-Ir,/bru/bru.r -Iq,/bru/bru.q'
DEVS='-f tape0 -f tape1'
bru -c $IOPS $DEVS $FILES 2>/etc/dailybru.log
```

where tape0 and tape1 are tape units that the operator loads before leaving.

Every day at 2:30 a.m., cron will run the /etc/dailybru script, causing BRU to begin backing up all files and subdirectories under /usr. BRU will continue to run until it finishes or until it hits the end of the tape in tape1, at which point it will wait for the operator to communicate with it via the fifos /bru/bru.r and /bru/bru.q. The file /etc/dailybru.log will contain any warnings or error messages issued by BRU.

This use of device cycling allows BRU to use multiple devices without the operator's presence being required. When BRU runs out of space on the first volume, instead of the operator (who has gone home) having to physically mount another volume, BRU automatically switches to a different volume, physically mounted elsewhere, to continue the archive creation operation.

Setting Up a cron Entry

Many UNIX system administrators like to schedule their backups to run automatically, often in the middle of the night. This can be accomplished by using the cron facility. cron executes commands specified in a cron table that is created by the user. This table can be listed or modified with the crontab command.

Here is an example of how to list the current contents of the cron table. The commands shown are typical entries for the root user:

```
# crontab -l
17 5 * * 0 /etc/cleanup > /dev/null
0 2 * * 0,4 /usr/lib/cron/logchecker
0 3 * * * /usr/lib/cleantmp > /dev/null
20 1 * * * /usr/bin/calendar -
```

Each line in the crontab consists of six fields. The first five determine the time. The sixth field is the command that will be executed. The six fields are interpreted as follows:

Field	Description
1	minutes (0 to 59)
2	hour (0 to 23)
3	day of month (1 to 31)
4	month (1 to 12)
5	day of week (0 to 6, 0=Sunday)
6	the command to execute

Some sample crontab entries are described below.

Run BRU every day at 2:30 a.m.:

```
30 2 * * * /bin/bru -c /
```

Run BRU at 10:00 p.m., Monday through Friday:
```
0 22 * * 1-5 /bin/bru -c /
```

Run the fullbru command at 3:00 a.m. every Saturday:

```
0 3 * * 6 /bin/fullbru
```

Run the incbru command at 4:00 a.m., Tuesday through Friday:

```
0 4 * * 2-5 /bin/incbru
```

Updating the crontab involves three steps:

    1: Create a file that contains a copy of the current cron table.
    2: Edit the file (or simply append a line to add a new entry).
    3: Update the cron table with the edited file.

Here's a simple example that will add a new entry to the cron table:

```
# crontab -l > cron.lst
# echo "30 2 * * 2-6 /bin/bru -c" >> cron.lst
# crontab cron.lst
```

To verify that the entry has been added, list the contents of the cron table again:

```
# crontab -l
17 5 * * 0 /etc/cleanup > /dev/null
0 2 * * 0,4 /usr/lib/cron/logchecker
0 3 * * * /usr/lib/cleantmp > /dev/null
20 1 * * * /usr/bin/calendar -
30 2 * * 2-6 /bin/bru -c
```

Note that the new entry for BRU has been added. For more information on creating crontab entries, refer to your system documentation on cron and crontab.

BRU's I/O Streams

BRU has several I/O streams that it uses for providing information to the operator and for accepting input when such input is required. The names of the streams used internally are:

stderr

This is the stream to which all error and warning messages are written, when such messages are **NOT** part of a query to the user to which a response is required. For example, failure to open a particular file for archiving because of permissions would generate an error message to the stderr stream.

logfp

This is the stream to which all information generated as a result of the –v option is written. Also, output as a result of the –h option is written to logfp. If the user specifies a –ll option to BRU, then logfp is opened using that file name. Otherwise, logfp is connected to stdout, unless the archive is being written to stdout, in which case logfp is switched to stderr.

ttyout

This is the stream to which all messages are written that are part of a query to the user for input, to which a reply is required before execution continues. Execution is blocked until a reply is received (see ttyin). When BRU is run in the foreground, ttyout is connected to the user's terminal by explicitly opening the file specified by the –Iq option to BRU. If there is no –Iq option specified, then /dev/tty is tried. If this open fails, then ttyout is connected to stderr.

ttyin

This is the stream from which replies are read in response to queries posted to the ttyout stream. Execution is blocked until the reply is read from the ttyin stream. When BRU is run in the foreground, ttyin is connected to the user's terminal by explicitly opening the file specified by the –Ir option to BRU. If there is no –Ir option specified, then /dev/tty is tried. If this open fails, then ttyin is connected to stderr.

Handling Sparse Files: –S threshold

With the UNIX filesystem, it is possible to create files which take up much less space in the filesystem than their size would indicate. These files are generally known as sparse files, and commonly occur in database or scientific applications. A sparse file can be loosely defined as one in which large areas of null bytes are created by seeking to a particular file offset before writing any actual data. The following example shows creation of a sparse file using the UNIX dd command to create an empty file, seeking to an offset of 1M in the file, and then writing a string at that offset:

```
$ df
/   /dev/dsk/c0d0s0   20330 blocks   9670 i-nodes
/x  /dev/dsk/c5d0s0   13242 blocks   7168 i-nodes
$ echo "End Of File" | dd of=sparsefile bs=1k seek=1k
0+1 blocks in
0+1 blocks out
$ ls -l sparsefile
-rw-rw-r— 1 fnf sys 1048588 Oct 5 09:50 sparsefile
$ od -c sparsefile
0000000 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
*
4000000 E n d   O f   F i l e \n
4000014
$ df
/   /dev/dsk/c0d0s0   20330 blocks   9670 i-nodes
/x  /dev/dsk/c5d0s0   13236 blocks   7167 i-nodes
```

Note that the free space in the /x filesystem, where the sparse file was created, was 13242 blocks before creating the 1M sparse file, and 13236 blocks afterwards. Thus the sparse file actually uses only 6 disk blocks (each 512 bytes), or 3K bytes of actual disk space.

Now look what happens when we copy the sparse file to another file:

```
$ df
/   /dev/dsk/c0d0s0   20330 blocks   9670 i-nodes
/x  /dev/dsk/c5d0s0   13230 blocks   7167 i-nodes
$ cp sparsefile bigfile
$ df
/   /dev/dsk/c0d0s0   20330 blocks   9670 i-nodes
/x  /dev/dsk/c5d0s0   11170 blocks   7166 i-nodes
$ ls -l sparsefile bigfile
-rw-rw-r— 1 fnf sys 1048588 Oct 5 09:50 sparsefile
-rw-rw-r— 1 fnf sys 1048588 Oct 5 10:04 bigfile
$ od -c bigfile
0000000 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0 \0
*
4000000 E n d   O f   F i l e \n
4000014
```

The thing to note here is that when we copied the file, the process of copying the file did not preserve the sparseness of the file. The copy of the file named bigfile now actually takes up about 1M of disk space (2060

blocks). Note that BRU can be used to copy the file while preserving the sparseness, as given in an example earlier. From the viewpoint of a user process reading or writing the file, both files look identical, as is shown by the dump of the file using the UNIX od command. However, the original sparse file only uses 6 disk blocks and the copy uses 2060, quite a difference! BRU has a special option, the –S option, which turns on features that reduce the amount of archive space used by sparse files, and that preserve the sparseness of the file when it is extracted from the archive.

Using Remote Devices (non-OS X)

BRU incorporates a special feature that is similar to that used in the remote dump rdump and remote restore rrestore programs. Instead of specifying the name of a local device to use as the archive, you can give the name of a device on another system connected to the local system via some high speed network such as ethernet.

In the example below the local system, the one running BRU is named myhost. The remote system, the system with the desired archive device attached is called remotehost, and the remote device is known as /dev/rmt0. To backup the files from the local system "myhost" and store them on the remote system "remotehost", /dev/rmt0 device, you would run the command below on the local host "myhost":

```
bru -c -vv -f remotehost:/dev/rmt0
```

To restore files from the same remote system you would run this command:

```
bru -x -vv -f remotehost:/dev/rmt0
```

You should also create an entry for the remote device in myhost's brutab file. As long as you have the appropriate programs and permissions to be able to run a remote shell on remotehost, and you have the remote device server program /etc/rmt on remotehost, then BRU should be able to read and write the remote device /dev/rmt0. Note that some versions of the /etc/rmt program have a maximum I/O buffer size of 10K, which is smaller than the default I/O buffer size used by BRU. If you get an error using remote devices, try setting your I/O buffer size to 4K using the –b option. Using remote devices can sometimes be very slow. Once you have established that you can successfully read and write a remote device, you might try the double buffering option to increase the speed of remote reads and writes. **NOTE**:

The /etc/rmt program is not furnished with certain versions of UNIX. If you cannot find a copy (source code can be found at many Internet sites), please contact TOLIS Group. A complete source code listing (and full explanation) can also be found in the following book for C programmers: UNIX Network Programming, W. Richard Stevens, Prentice-Hall, Englewood Cliffs, NJ, 1990, ISBN 0-13-949876-1.

Restoring Shared Library Files - SmartRestore

Many newer versions of UNIX (like System V, Release 4) use shared library files. These files contain common C-language routines that are called by most UNIX programs. Nearly every UNIX program uses the libc shared library, so this file is always active. Shared library files are normally stored in one of the library directories like /lib, /shlib or /usr/lib. Shared library filenames are often distinguished by a special suffix like "_s" or ".so" or ".so.1" (for example, /shlib/libc_s or /usr/lib/libc.so.1).

WARNING: Serious problems can occur if you try to restore (overwrite) an active shared library file-usually your system will crash! BRU does its best to be intelligent about the way it handles restoring these types of files through the SmartRestore functionality. If a file is in use (text file busy), BRU will attempt to relocate the existing file before restoring the copy in your archive. The text busy flag and memory pointers will follow the move to the file's new location, so your system will continue to operate. However, the old version of the in-use file will still be on the system when you next reboot.

When restoring using the SmartRestore process, BRU will examine the contents of the /etc/brusmartrest file for specific files to handle this way even if the file's text busy flag is not set. Example entries for the /etc/brusmartrest file are:

```
s  *.so    # Be careful with shared libraries
s  *.so.*
s  */bru   # don't overwrite the bru program
```

When BRU restores a file that matches one of these specifications, the original file is moved to /usr/tmp/ and named DELETE.XXXXXX where the X's are numbers. A shell script is created called /bru/bruremovelog that serves as both a log file of the files restored using this method and a simple script to delete the moved original files once

the system is rebooted. Another way to avoid problems when restoring files, is to exclude any active shared library files using the –X option with appropriate entries in the /etc/bruxpat file. Here are two examples of exclusion patterns that can be placed in your bruxpat file:

For Mac OS X:
```
xs *.dylib
xs *.so
xs */bru
```

For System V, Release 4:
```
xs *.so.*
xs *.so
xs */bru
```

For SCO UNIX:
```
xs /shlib/*_s
xs */bru
```
These patterns may not be correct for your system or backup method. Please refer to your system documentation for details on the location of any shared libraries on your system.

Appending to Existing Archives

Tape drives allow you to write several archives, or filesets on a single tape. To do this, you must prevent the tape from performing a rewind after it has completed writing (the normal mode of operation). This can be accomplished by using the norewind device node. When the norewind device is used, BRU will begin to write data at the current tape position and will stop the tape, without rewinding, when the backup has completed. Be aware that when the norewind device is used, it is impossible to rewind the tape and BRU's automatic verification features will be disabled.

If you use the norewind device, you must position the tape at the proper file mark before attempting to read or write. Tape utilities that do this are normally furnished with your UNIX system (or can be furnished by your tape drive vendor). The names of these tape utilities vary widely, but common names are: tape, tapecntl and mt. Refer to your system documentation for specific details. **NOTE**: You should not attempt to store multiple archives on a single tape unless you have very good reasons for doing so. Do not do it simply to save on tape costs.

Many newer tape drives can handle several Gigabytes and it may seem appealing to have the capacity to store a week's worth of backups on a single tape. Do this, and you're "putting all your eggs in one basket." You could lose all your backup data if your tape is damaged. The cost of a few more tapes will seem cheap compared to the trouble you'll have trying to restore (or re-create) your data.

Live System Backups

When backing up files on a UNIX system, it is always best if the system is quiet and no file changes are occurring. That's why many system administrators choose to run their backups late at night.

On some busy UNIX systems, especially those with active database programs, file changes occur 24-hours a day. If files are changing, BRU will attempt to archive them, but errors may occur. If a file changes while BRU is reading it, a warning message will normally be issued. Other errors may occur if BRU is attempting to compress a file that is changing. In either case, the data that was archived may not be correct. In fact, it may contain serious errors.

BRU does not perform any file-locking, so there is no way for BRU to stop a file from changing during a backup. In most cases, file-locking would be worthless anyway, especially for database files. This is because BRU has no way of determining if a file is part of a changing database or of knowing which groups of database files should be locked. Theoretically, BRU could simply lock each file individually, but this would probably cause the database program to crash. Most database programs have complex locking schemes and do not expect anyone else to be locking their files.

Fortunately, some of the database vendors have recognized the problems in trying to do hot backups and furnish special commands that allow you to temporarily "freeze" the database files while a backup is done. Any changes made to the database during the backup will be written to a special temporary area. Once the backup is done, all the changes will be applied to the database files. The following example illustrates how to put an Oracle database into backup mode, perform the backup with BRU, and then return the database to normal.

First, issue SQL commands to "freeze" the Oracle database:

```
connect internal
alter tablespace name begin backup
exit
```

Now run BRU and back up the database files:

```
bru -cv /database
# After the backup is done, "unfreeze" the database:
connect internal
alter tablespace name end backup
exit
```

The above database commands are shown only as an example. They may not work correctly for your particular database. For additional information, refer to your database documentation or contact your database vendor.

Appendix A - Files That Are Installed

Overview

The BRU files that are installed on your system are listed below. For simplicity, we often refer to a tape drive rather than listing all the possible media types. Where we have done this, it should be understood that all media types are included.

Main Application:

/Applications/ArGest Backup.app
The macOS ArGest Backup UI application.

/opt/ArGest Backup/
The Linux ArGest Backup UI application.

C:\Program Files\ArGest Backup
The Windows ArGest Backup UI application.

Environment – catalogs and settings:

/Library/Application Support/ArGest Backup/
The binary, catalog, and settings are stored in this folder on macOS systems.

/usr/local/ArGest Backup/
The binary, catalog, and settings are stored in this folder on Linux systems.

C:\ProgramData\TOLIS Group\ArGest Backup
The binary, catalog, and settings are stored in this folder on Windows systems.

The following configuration files are located in the **etc** folder within the environment folders listed above:

etc/brutab
The parameter file that customizes BRU for your system. You will need to further customize this file by entering the parameters for your tape drive.

etc/bruxpat
A file in which you may specify patterns that will match files you want to include in or exclude from your backups and restores. The **bruxpat** file permits BRU to archive and extract files selectively. See Appendix J, "The **bruxpat** File."

Other files:

etc/brusmartrest
This file contains information regarding to restoring open or shared files that require special handling such as the system kernel or shared libraries that are in use.

/usr/local/bin/bru
The bru backup and restore program on Linux and macOS. The Windows copy is stored within the ArGest Backup environment folder listed above.

/usr/local/bin/incbru
A shell script that uses BRU to perform an incremental backup. The **incbru** command will speed daily backups by causing BRU to back up only files that have been changed since the last time **fullbru** was run. You may want to customize the **incbru** script to handle your specific backup needs. See Appendix H, "The Shell Scripts **incbru** and **fullbru**."

/usr/local/bin/fullbru
A shell script that will perform a full system backup (level 0) as part of an incremental backup procedure. The **fullbru** script works with **incbru**, described above.

/usr/local/bin/brutalk
A program used to communicate with BRU when BRU is run in background mode. See Appendix I, "The **brutalk** Program."

/Library/Application Support/ArGest Backup/README

A file that contains specific information for your type of system. It also contains information that may have been too late to be included in this manual.

On Linux and Windows systems, additional files and utility programs may also be installed in the **bru** directory at the root of your system drive. These are optional and are not needed to make BRU or ArGest Backup work properly. This directory also contains a copy of the **bru** executable as **/bru/bru or C:\bru\bru.exe**. This is not the version that is normally executed. This copy is stored as a safety precaution, in case the executable copy, stored as **/usr/local/bin/bru**, is lost or corrupted.

Appendix B - Tape / Archive / Catalog Design Notes

There are 2 folders that are involved in containing the ArGest Backup catalog and tape info:

Tapes: Holds info about known tapes
Catalogs: Holds info about known archives

These folders contain either plain text files or the gzip compressed catalog data.

Archive and Tapes folder content explanation

You run a backup and create a new archive to a new tape that fits on one tape, Archive ID 4987e2342205. The following entries are created:

```
Tapes/4987e2342205.1
Catalogs/4987e2342205
Catalogs/4987e2342205/arLabel
Catalogs/4987e2342205/4987e2342205_Run.log
Catalogs/4987e2342205/4987e2342205_metadata.xml
Catalogs/4987e2342205/BRU-4987e2342205.gz
Catalogs/4987e2342205/media
```

Examining the Tapes file, you see:

```
name=The Name You assigned the tape
barcode=TG1001L6
4987e2342205.1 0
```

This indicates that the tape `4987e2342205.1` contains the first segment of archive `4987e2342205` and that segment starts on block 0 of the tape.

If you look at the media file in that archive's folder, you see:

```
4987e2342205.1 0
```

You also see the archive label that you assigned in the arLabel file for the archive. Remember, the archive label is not the same as the tape set(s) name.

If you then append a new archive to that tape that still fits on the same tape, Archive ID `4987e7ee5008`. The append starts at block 45334 on the tape. When you look at the Tapes folder, you still only see the one file:

```
Tapes/4987e2342205.1
```

By examining the file you will see the new, appended archive:

```
name=The Name You assigned the tape
barcode=TG1001L6
4987e2342205.1 0
4987e7ee5008.1 45334
```

In the Archives folder, you now have a new folder for the new archive:

```
Catalogs/4987e7ee5008
Catalogs/4987e7ee5008/arLabel
Catalogs/4987e7ee5008/4987e7ee5008_Run.log
Catalogs/4987e7ee5008/4987e7ee5008_metadata.xml
Catalogs/4987e7ee5008/BRU-4987e7ee5008.gz
Catalogs/4987e7ee5008/media
```

In this case, when you examine the new archive's media file, you'll see the original archives ID plus the offset for our new archive:

```
4987e2342205.1 45334
```

Again, the single entry tells us that this new archive also fit onto the one tape.

The exceptions would be if an archive spreads across multiple tapes. In that case, there would be multiple files in the Tapes folder named for the archive ID of the first archive and appended with the volume number. So a new archive `49ae5563e087` that

extended over three tapes would have three entries in the Tapes folder:

```
Tapes/49ae5563e087.1
Tapes/49ae5563e087.2
Tapes/49ae5563e087.3
```

If you examined each of those files, you would see the name= field followed by the archive ID with the volume number and the offset of 0 since each segment starts at the beginning of the tape.

```
Tapes/49ae5563e087.1
      name=My New Multitape archive
      barcode=TG1001L6
      49ae5563e087.1 0

Tapes/49ae5563e087.2
      name=My New Multitape archive
      barcode=TG1002L6
      49ae5563e087.2 0

Tapes/49ae5563e087.3
      name=My New Multitape archive
      barcode=TG1003L6
      49ae5563e087.3 0
```

If you then checked the media file in the Archives folder for the new archive, you would find:

```
Catalogs/49ae5563e087/media
      49ae5563e087.1 0
      49ae5563e087.2 0
      49ae5563e087.3 0
```

This indicates that the archive covers 3 tapes.

Finally, if you appended another archive to the end of that third tape, You would not add another Tapes file, but rather you would update the existing `49ae5563e087.3` file to include the info on the appended archive:

```
Tapes/49ae5563e087.3
      name=My New Multitape archive
      barcode=TG1003L6
      49ae5563e087.3 0
      49ae5871e344.1 134232
```

Tape Layout

As shown in the figure below, the BRU tape layout is basically 2 elements -

• The Backup Set for a given job or system
• The Catalog for that Backup Set

BRU Tape Layout

Each of these segments is separated by a 'filemark' (FM) - a special block type on tape that allows the drive to move easily and quickly between data segments. A tape drive can search for filemarks without reading the intervening data blocks. This means that an LTO tape can locate a file on tape 200 times faster than an equivalent read of the intervening data blocks. The tape drive determines the end of data (EOD) by placing two filemarks just before the null space.

To access a specified segment, use tapectl to step to its starting point. For example, if you wanted to extract (restore) the data in the second backup set illustrated above, you would step over 2 filemarks with tapectl as in:

```
      tapectl -f ntape0 fsf 2
```

This will move the tape forward to the start of the 3rd data segment on the tape (BACKUP SET 2) leaving the tape positioned ready to read the data in that data segment.

For more information on using tapectl, please refer to The **TOLIS Tape Tools** manual that is included with your copy of ArGest Backup.

ArGest Backup Catalog Format

If you would like to make use of BRU's QFA functionality, you may do this outside of the BRU GUI using simple command line processes. ArGest Backup writes the catalog info in a format that looks like this:

```
VL:c|1019904|1|106496|3984|/Users/bruman/Agent/lzo.dll
```

The fields (separated by the "|" symbol) are defined as follows (all numbers are in decimal):

Field	From Example Above	Description
1	VL:c	Verbosity Line, mode is create
2	1019904	Offset within the archive for this file
3	1	Volume (tape) this is on (if multi-tape)
4	1016496	File Size (Bytes)
5	3984	QFA Tape Block Number
6	/Users/bruman/Agent/lzop.dll	File path and name

Command Line Restore Example using QFA

To set up a scenario, imagine that we have a tape with our entire system backed up via ArGest Backup with the archive ID of 3d4977f008d8. We need to restore the /Users/Shared/ folder.

Here are the steps we'll use to determine the location of that folder in the tape archive (QFA location) and restore it:

```
# cd /Library/Application\ Support/BRU\ PE
# cd Catalogs/3d4977f008d8
# zgrep -a \/Users\/Shared BRU-3d4977f008d8,gz | head -1
VL:c|0|1|816|65241|/Users/Shared/
```

In this case, the QFA location for /Users/Shared is at tape block 65241. If we did not use QFA to start at that location on the tape, BRU would need to start the restore process at the beginning of the archive on tape and read through all of the intervening data to reach the point where the data we are looking for is located. By using the tape drive's "seek" capability, we can pre-position the tape to the start location of our data at 120x to 200x normal read speed. This reduces the time required to restore a specific data set dramatically in a very large archive.

By using tapectl, we can position the tape ready to restore:

```
tapectl -f ntape0 seek 65241
```

The tape will move and stop at the specified block location.

To restore the data, overwriting any data that may be on your disk, we simply issue the following BRU command:

```
bru -xvvf ntape0 -QV -ua /Users/Shared
```

(refer to Chapter 17 for detailed information on the arguments used).

Appendix C - The BRU Help Command

Overview

This section describes BRU's two help options, **-h** and **-hh**. The first prints an abbreviated manual page for quick reference. The second lists brutab parameters and their current values.

Using bru with the –h Option

This is the mini-manual page that is printed when BRU is invoked with the **-h** (help) option. It is intended to be a concise reminder of the various options to BRU and their meanings. This sample page may not be identical to the version printed by your particular copy of BRU. Generally, this is a duplication of the BRU **manpage**. Please refer to "Appendix C: The BRU Manual Page."

Using bru with the –hh Option

BRU may also be invoked with the **-hh** option. This version of the command looks up all the parameters for the default device and prints their current values. It is a good way to verify parameter values after they have been changed. The **-hh** option has the following variant: It may be followed by –f and a device name. For example:

```
# bru -hh -f ntape0
```

In this case, the command will print the current parameter values for the named device. This command is used mostly to verify that BRU is reading the proper values from the **brutab** file. Invoking bru with **-hh** produces a listing similar to the following:

```
**** environment variables / global brutab parameters
****
BRUTABONLY=NO
DEVNAMECHECK=NO
MATCHLEVEL=2
MAXFILENAMELEN=512
MAXWRITES=1000
OVERWRITEPROTECT=YES
READCHECKLEVEL=1
RECYCLEDAYS=1
ZBUFSIZE=1048320
ZINBUFSIZE=4096
DIRDATESELECT=NO
ZOUTBUFSIZE=6144
MOUNTCMD="(null)"
UNMOUNTCMD="(null)"
BRUHELP="/bru/bruhelp"
BRUMAXWARNINGS=1000
BRUMAXERRORS=500
BRUXPAT="/etc/bruxpat"
BRURAW="/etc/bruraw"
BRUSMARTREST="/etc/brusmartrest"
BRUREMOVELOG="/usr/adm/bruremovelog"
BRUTMPDIR="/usr/tmp"

 **** brutab parameters for device: ntape0 ****

devname="EXABYTE Tape"
tape: ON
rawtape: ON
reblocks: OFF
ignoreclose: OFF
noautoscan: OFF
rawfloppy: OFF
shmcopy: ON
format: OFF
eject: OFF
qfwrite: OFF
reopen: ON
noreopen: OFF
norewind: OFF
advance: OFF
openrw: OFF
largefile: OFF
resetcmd="(null)"
fmtcmd="(null)"
```

```
rfmcmd="(null)"
wfmcmd="(null)"
retencmd="(null)"
rewindcmd="(null)"
positioncmd="(null)"
locatecmd="(null)"
ejectcmd="(null)"
openretries=1
iotimeout=0
iowait=0
opentimeout=3600
maxrewindtime=0
minrewindtime=25
size=1952000K
bufsize=20480
asbufsize=262144
maxbufsize=0
minbufsize=0
blocksize=0
endnulls=0
shmseg=7
shmmax=204800
shmall=0
seek=0
prerr=0
pwerr=0
zrerr=0
zwerr=0
frerr=0
fwerr=0
wperr=5
ederr=0
rmtsh="no-rsh"
rmtsvr="/etc/rmt"


**** system info ****
compile date: Wed Feb  1 14:32:00 1995
compiled on: odt20 odt20 3.2 2 i386
```

Using bru with the –hhh Option

BRU, when run with this option will produce a list of the error code definitions recognized by BRU.

```
**** error code definitions ****
EPERM=1
ENOENT=2
ESRCH=3
EINTR=4
EIO=5
ENXIO=6
E2BIG=7
ENOEXEC=8
EBADF=9
ECHILD=10
EAGAIN=11
ENOMEM=12
EACCES=13
EFAULT=14
ENOTBLK=15
EBUSY=16
EEXIST=17
EXDEV=18
ENODEV=19
ENOTDIR=20
EISDIR=21
EINVAL=22
ENFILE=23
EMFILE=24
ENOTTY=25
ETXTBSY=26
EFBIG=27
ENOSPC=28
ESPIPE=29
EROFS=30
EMLINK=31
EPIPE=32
EDOM=33
ERANGE=34
EWOULDBLOCK=90
EINPROGRESS=91
EALREADY=92
ENOTSOCK=93
EDESTADDRREQ=94
```

```
EMSGSIZE=95
EPROTOTYPE=96
ENOPROTOOPT=118
EPROTONOSUPPORT=97
ESOCKTNOSUPPORT=98
EOPNOTSUPP=99
EPFNOSUPPORT=100
EAFNOSUPPORT=101
EADDRINUSE=102
EADDRNOTAVAIL=103
ENETDOWN=104
ENETUNREACH=105
ENETRESET=106
ECONNABORTED=107
ECONNRESET=108
ENOBUFS=63
EISCONN=110
ENOTCONN=111
ESHUTDOWN=112
ETIMEDOUT=114
ECONNREFUSED=115
ELOOP=150
ENAMETOOLONG=78
ENOTEMPTY=145
ENOMSG=35
EIDRM=36
ECHRNG=37
EL2NSYNC=38
EL3HLT=39
EL3RST=40
ELNRNG=41
EUNATCH=42
ENOCSI=43
EL2HLT=44
EDEADLOCK=56
```

Appendix D – The BRU Manual Page

Overview

The full BRU manual page follows. Appendix B is an abbreviated form of this manual page. It contains a help list of bru options and short definitions. It also shows you how to list brutab parameter values and device information. Appendix D contains a table of options cross referenced to the bru modes with which they may be used.

```
NAME

    bru - backup and restore utility

SYNOPSIS

    bru modes [control options] [selection options] [files]

DESCRIPTION

    Bru is a UNIX filesystem backup utility with significant
        enhancements over other more common utilities
        such as tar, cpio, volcopy, and dd. Some of bru's
        capabilities include:

o   Full or incremental backup with quick and easy
        restoration of files.

o   Multiple physical volumes per archive.

o   Data integrity assurance via checksum computation on
        every archive block.

o   Ability to save and restore directories, symbolic links,
        block special files, and character special files.

o   Comparison of archives with current directory hierarchy.

o   Ability to recover files from corrupted archives or
        damaged media with minimal data loss.

o   No inherent maximum archive buffer size.

o   Improved performance through random access archive I/O
        when available.

o   Automatic byte or half word swapping as necessary when
        reading archives produced on other machines.

o   Recognition of filename generation patterns in the same
        form as the shell for files read from an archive.

o   Intelligent handling of large, sparse files.

    When files are specified on the command line then the
        actions to be performed are limited to those
        files. If a named file is a directory then it and
        all its descendants are used. If no files are
        specified then the default for writing archives
        is all files in and below the current directory.
        The default for reading archives is selection of
        all files in the archive.
```

```
    If "-" is given instead of files then the standard input
        is read to obtain the file list. This is useful
        in conjunction with the find command to provide
        finer control over files selected for backup.
        Obviously this mode is only valid when bru is not
        also reading its archive from the standard input.

DEFAULTS

    Various default parameters, such as archive device name
        and size, archive buffer size, controlling
        terminal name, etc., are system dependent. These
        defaults, along with version, variant, and other
        miscellaneous internal information, may be
        discovered via the -h mode.

MODES

    One or more of the following modes must be specified. The
        order of execution, from highest priority to
        lowest, is ecitxdgh.

-c  Create a new archive. Forces a new archive to be created
        regardless of whether one currently exists.
        Writing starts at the first block.

-d  Differences between archived files and current files are
        detected and reported. May be specified more than
        once, as -dd, -ddd, or -dddd, to control level of
        difference checking.

    When specified as -d, bru reports when it discovers that
        a regular file's size (st_size) or contents (when
        compared as byte streams) has changed since the
        archive was made.

    When specified as -dd, bru reports additional differences
        in modification date (st_mtime) access mode
        (st_mode), number of links (st_nlink) for non-
        directory files, differences in the contents of
        symbolic links, owner id (st_uid), and group id
        (st_gid).

    When specified as -ddd, bru reports additional
        differences in host device (st_dev), major/minor
        device (st_rdev) for special files, and time of
        last access (st_atime) for regular files.

    When specified as -dddd, bru reports all differences
        except time of last status change (st_ctime is
        not capable of being reset), major/minor device
        numbers for non-special files (meaningless), and
        size differences for directory files (may have
        empty entries). The -dddd mode is generally
        meaningful only during a verification pass with
        full backups of quiescent filesystems.

-e  Estimate media requirements for archive creation with
        same arguments. Prints estimated number of
        volumes, number of files to be archived, total
        number of archive blocks, and total size of
        archive in kilobytes. If the media size is
```

unknown or unspecified, it is assumed to be infinite. Estimates are for uncompressed data only; the -Z option will be ignored.

-g Dump archive info block in a form more easily parsed by programs implementing a complete filesystem management package. Performs no other archive actions.

-gg List archive list of files created during the bru -cvG mode. This has been deprecated and is no longer used in BRU 17.0.

-h Print help summary of options. Also prints some internal information such as version number and default values for archive pathname, media size, archive buffer size, etc.

-i Inspect archive for internal consistency and data integrity. When -vv option is also given, prints information from archive header block.

-t List table of contents of archive. When used with the -v option, will give a verbose table of contents in the same format as the "ls -l" command. When used with the -vv option, will also indicate what files are linked to other files and where symbolic links point.

-x Extract named files from archive. If an archived file is extracted (see -u option) then the access mode, device id (special files only), owner uid, group uid, access time, and modification time are also restored. If the -C flag is given (see below), then the owner uid and group uid will be changed to that of the current user.

Nonexistent directories are recreated from archived directories if possible. Otherwise they are created with appropriate defaults for the current user. Extracted or created directories are initially empty.

CONTROL OPTIONS

Many of the control options are similar in function to their tar or cpio equivalents.

Sizes are specified in bytes. The scale factors G, M, k, or b, can be used to indicate multiplication by 1024*1024*1024 (one Gigabyte), 1024*1024 (one Megabyte), 1024, or 2048 (the size of a bru tape block) respectively. Thus "10k", "5b", and "10240" all specify the same number of bytes.

-a Reset the access times of disk files that have been read while performing other actions.

-b bsize

Use bsize as the archive input/output buffer size. The minimum is the size of an archive block (2k or 2048 bytes) and the maximum is determined by

available memory and I/O device limitations. If bsize is not an even multiple of 2048 bytes, it will be rounded up. Normally this option is only required with -c mode since bru writes this information in the archive header block. If specified, bsize overrides any existing default value (generally 20k), whether built in or read from the archive header.

-B Useful in shell scripts where bru is run in the background with no operator present. Under these conditions, bru simply terminates with appropriate error messages and status, rather than attempting interaction with the terminal.

-C Change the owner (chown) and group of each extracted file to the owner uid and group gid of the current user. Normally, bru will restore the owner and group to those recorded in the archive. This flag causes bru to follow the system default, with extracted files having the same owner and group as the user running bru, including root. (Under 4.2 BSD, the default group is that of the directory in which the file is created.)

The -C option is useful with archives imported from other systems. In general, it should not be used by the operator or system administrator when restoring saved file. Use the -tv option to see the owner and group of files stored in the archive.

-D Causes bru to use double buffering to the archive device on systems that have System V-style shared memory. Depending on hardware constraints, double buffering may dramatically increase the archive device I/O rate but may adversely affect the error recovery algorithms.

-f path

Use path as the archive file instead of the default. If the path is "-" then bru uses the standard input for archive reading or standard output for archive writing, as appropriate.

If multiple -f options are given, each path is added to a list of files to cycle through each time a volume change is required. When the end of the list is reached, bru automatically cycles back to the first path and waits for confirmation to continue the cycle again. Any input other than a carriage return will cause bru to use the newly entered path and to abort the cycling for the remainder of the current run.

-F Checksum computations and comparisons are disabled. This mode is useful when the output of one bru is piped to the input of another bru, or when the data integrity of the archive transmission medium is essentially perfect. Archives recorded in this

mode must also be read in this mode. Be aware
that some of the automatic features of bru, such
as automatic byte swapping and AUTOSCAN, are not
functional in fast mode.

-Iqrlb

> When BRU is running in the background, there is no way to
> interact with it. The -I qrlb options allow a
> users to setup a set of read and write fifos that
> a second program BRUTALK and work with to
> responed with BRU. Below we have listed the
> "qrlb" options that are currently valid with the
> -I option.

q,fifo write queries to fifo

r,fifo read responses from fifo

l,file write log info to file

b Use the default fifos created durning install.

> /bru/bru.q (Default send fifo)

> /bru/bru.r (Default reply fifo)

The q and r options are useful for interacting with bru
> when it has been run without a controlling
> terminal. See the discussion under RUNNING FROM
> CRON.

-L str (c)

In create mode, label tape with given string (63 char
> max).

-L file

In create mode, the first 63 characters in file are used
> as the label for the archive members. In extract
> mode the first 63 characters within file are used
> for the comparison as described below.

-L str (x)

In extract mode, only restore if the label of the archive
> EXACTLY matches the given string. If the string
> does not match on the initial tape, the operation
> aborts. If the label does not match on
> subsequent tapes, a warning is issued, but the
> extraction continues.

-r rawdev

This enables BRU to backup or restore raw data
> partitions. A BRURAW file must exist and contain
> entries that define the raw data to be accessed.
> See the discussion of the BRURAW global variable
> below for information on the BRURAW file and
> location. The file contains entries in this
> format:

Raw Device Name   Size   Blk Size Starting Offset

———-   —   ——   ———-

a sample entry would be:

/dev/rdisk0   30M   512   0

An entry MUST exist in the BRURAW file or bru will abort
> the operation with an error message.

When backing up raw partitions, you must either have the
> device node in the tree relative to your current
> location or you must explicitly declare the raw
> device on the command line as a file name. For
> example, if you are in / and the raw device node
> is /dev/rdisk1, you can use this command line to
> back it up properly:

bru -cvV -r /dev/rdisk1

If, however, you are invoking the bru command from your
> home directory, you must include the path to the
> device node on the command line:

bru -cvV -r /dev/rdisk1 /dev/rdisk1

Additionally, the filesystem being backed up MUST be
> locked or unmounted to prevent any writes from
> occurring during the backup. If this is not
> done, the data could be corrupted during the
> process of backing it up.

-T file

Translate on restore - rename or relocate files based on
> the contents of a translation file. The file can
> be any file name. It is an ASCII text file which
> contains two columns - all files that contain the
> text in column 1 will have that text translated
> to the text in column 2.

* It is important to note that there can be NO empty
> fields in this file. You MUST provide both
> columns for each line in the translation file.

This translation applies to directories, names and
> extensions. By default, symbolic links will not
> be translated. See the -Q option below.
> Translate works with all BRU modes except create
> (-c).

To restore all of the files in /home/bob to /home/ted,
> you should have the following information in your
> file.:

> /home/bob   /home/ted

No further interaction is required by the user. Also
> multiple translations can occur within a single
> restore. We could perform both of the above

translations and others by placing the columns into a single translation file as follows:

/home/bob   /home/ted

/usr/lib   /usr2/lib

/home /u2

...etc

The translation file can contain as many translation lines as necessary, but each line must consist of a balanced pair of entries.

-Q HLRSV

Changes default BRU operation as described below:

HThis turns off placing small files into tape headers. Using this option allows you to create BRU tapes that are compatible with versions of BRU prior to 14.2

LUse a literal string as a tape label. This overrides BRU's attempt to look for a file from which to read the tape label. This can be useful if you have a file that is the same name as the label you wish to apply.

RDisable SmartRestore. This turns off BRU's handling of open or shared files. It is not recommended that you override this setting.

STranslate Symbolic Links. Used in conjunction with the -T option (see above) will force the translation of symbolic links.

VIgnore "Incorrect Volume" warnings. When restoring from a multi-tape set or beginning a restore from other than the first tape in a set, use of this option will prevent the normal warning about having the incorrect volume and continue with the restore.

-l Ignore unresolved links. Normally bru reports problems with unresolved links (both regular and symbolic links). This option suppresses all such complaints.

-m Do not cross mounted file system boundaries during expansion of explicitly named directories. This option applies only to directories named in files. It limits selection of directory descendants to those located on the same filesystem as the explicitly named directory. This option currently applies only to the -c and -e modes.

-N level

Use level as the compression level if -Z is also specified. The default is to use level 3. The allowable values are 1 - 9, with 1 being lowest. The higher the level, the more agreesively BRU

will work to compress the file. This will be at the cost of processor power - higher compression uses more CPU power.

-O Overwrite the archive regardless of the settings for OVERWRITEPROTECT and RECYCLEDAYS.

-p Pass over files in archive by reading rather than seeking. Normally bru will use random access capabilities if available. This option forces reads instead of seeks. Use only on disk files or disk media.

-P flags

Pathname options that provide explicit control of expansion of directories, automatic archiving of parent directories, etc. Possible characters for flags are:

eDisable automatic expansion of explicitly named directories.

EEnable automatic expansion of explicitly named directories

fDisable filter mode. Builds internal tree of pathnames before doing anything with the pathnames from the input list.

FEnable filter mode. Each pathname from the input list is treated individually, and processing is done with that pathname before the next pathname is examined.

pDisable automatic archiving of parent directories of explicitly named files or directories.

PEnable automatic archiving of parent directories of explicitly named files or directories.

See the discussion under DIRECTORIES.

-R Remote files are to be excluded from the archive. If the system does not support remote filesystems, this option is ignored.

-s msize

Use msize as the media size. The effective media size will be computed from msize since it must be an integral multiple of the input/output buffer size (see the -b option). Normally this option is only required with the -c mode since bru writes this information in the archive header block. If specified, msize overrides any existing default value, whether built in or read from the archive header.

-S size

Enable options to deal more intelligently with sparse files (files with lots of null bytes). When used in conjunction with the -c mode, turns on automatic file compression for files that are

          larger than the specified size. When used in
conjunction with the -x mode, seeks will be used
to create blocks of null bytes in the output
file, rather than actually writing null bytes.
See the discussion under SPARSE FILES.

- -v Enable verbose mode. May be specified more than once, as
-vv, -vvv, or -vvvv, to get even more verbosity.
The -vvvv form of this option includes an
execution summary (see -V).

- -V Print execution summary only. The -vvvv option includes
an execution summary as part of its output.

- -X Apply exclusion patterns specified in the file /etc/
bruxpat (or in the file specified by the BRUXPAT
environment variable).

- -w Wait for confirmation. bru will print the file name and
the action to be taken and will wait for
confirmation. Any response beginning with 'y' or
'Y' will cause the action to complete. Any
response beginning with 'g' or 'G' will cause the
action to complete and will reset the -w option
so that no further confirmations will be
requested. Any other response will abort the
action.

- -Z Use lzop file compression. This is not the default
because not all versions of bru know how to deal
with compressed files. When the -v option is also
selected, the compression ratio for each file is
printed as a percentage. When this flag is used
in conjunction with the -t option on an archive
that contains compressed files, the actual
archive file sizes and names are printed, rather
than the original values before archiving.

          A limited amount of backward compatibility with non-
compressed versions of bru is provided. Archives
read by older versions will appear to contain
files that were precompressed prior to archiving.
The public domain compress utility can be used to
decompress such files after dearchiving. See the
-N option.

FILE SELECTION OPTIONS

          The file selection options control which files are
selected for processing. Note that some options
are valid only with specific modes.

- -n date

          Select only files newer than date. The date is given in
one of the forms:

Format	Example
DD-MMM-YY[,HH:MM:SS],[amc]	12-Mar-84,12:45:00.ac
MM/DD/YY[,HH:MM:SS],[amc]	3/12/84,ac
MMDDHHMM[YY],[amc]	0312124584,ac
pathname	/etc/lastfullbackup

          The time of day is optional in the first two forms. If
present, it is separated from the date by a
comma.

          By default, the modification and create times are used
for comparison. Other times can be used by
specifying letters after the date (a=access time,
m=modification time, c=create time). For example:

- -n 14-Apr-84,15:24:00,ac

          If the date is preceded by an exclamation point (!), then
files older than the specified date will be
selected.

          If date is the pathname of a file, then the modification
date of that file will be used. This is useful in
automated backups when a dummy file is "touched"
to save the date of the last backup. NOTE: do not
use the [amc] modifier with tis option.

- -o user

          Select only files owned by user. User may be specified
in one of three ways:

          As an ascii string corresponding to a user name in the
password file.

          As the pathname of a file, in which case the owner of
that file is used.

          As a numeric (decimal) value. This value will be the user
ID as found in the passwd file.

- -u flags

          When used in conjunction with -x mode, causes files of
the type specified by flags to be unconditionally
selected regardless of modification times.
Normally bru will not overwrite (supersede) an
existing file with an older archive file of the
same name. Files which are not superseded will
give warnings if verbose mode level 2 (-vv) or
higher is enabled. Possible characters for flags
are:

a	select all files (same as specifying all flags)
b	select block special files
c	select character special files
d	select directories
l	select symbolic links
p	select fifos (named pipes)
f	select regular files (same as r)
r	select regular files (same as f)

```
-U #Selection depth for backup or restore of files in
        relation to the current directory depth.  Files
        more than (#) levels Under this level will not be
        processed.
```

For example, if the current directory is /home/ted, -U0 (zero) will only backup the files and directory nodes in the home/ted directory. While the directory nodes will be backed up, any files in directories off of this level will not be backed up. With a level of 2, BRU will backup files in /home/ted/test, /home/ted/test/runone, /home/ted/test/runtwo, but not in /home/ted/test/runone/demo1.

This same feature works for restoring files and directory trees as well.

Selection of directories implies that only their attributes may be modified. Existing directories are never overwritten, this option merely allows their attributes to be set back to some previously existing state.

Selection of symbolic links implies that only the contents of the link will be modified. It is currently impossible under 4.2 BSD to change access time, modification time, or the file mode of a symbolic link.

EXAMPLES

Create (-c) a new archive of all files under "/usr/src", writing archive to file (-f) "tape0" using multiple tapes with a maximum size (-s) of 20 gigabytes per tape.

```
bru -c -f tape0 -s 20G /usr/src
```

Create (-c) a new archive on the default device in the first pass, archiving all files in and below the current directory which have been created or modified since 3 P.M. on 14-Jan-92 (-n). Then do a second pass to verify that there are no differences (-d) between the archive and current files. Each file is listed (-v) as it is processed.

```
bru -cvd -n 14-Jan-92,15:00:00
```

Archive all files owned (-o) by user "user1" using the default archive device.

```
bru -c -o user1 /
```

Copy a directory hierarchy from "/usr/u1" to "/usr/u2".

```
(cd /usr/u1; bru -cf - ) | (cd /usr/u2; bru -xf -)
```

Extract (-x) the regular file "/usr/guest/myfile" unconditionally (-ur) from an archive on file (-f) "tape0". Since the device size was recorded in the header block, it need not be specified. Note that option arguments do not need to be separated from their corresponding option flag by whitespace.

```
bru -x -ur –f tape0 ./usr/guest/myfile
```

Extract (-x) all C source files in "/usr/src/cmd" that have names beginning with characters 'a' through 'm'. Wait (-w) for confirmation before extracting each file.

```
bru -xw '/usr/src/cmd/[a-m]*.c'
```

Inspect (-i) a previously created archive on the default device, dumping the contents of the header block for inspection (-vvv) and verifying the internal consistency and data integrity of the archive.

```
bru -ivvv
```

Back up the entire root filesystem without crossing mounted (-m) filesystem boundaries. The archive will be written to file (-f) "tape0" using an I/O buffer size (-b) of 10k bytes. A record of all files processed will be written to file "brulogfile" for future reference.

```
cd /

bru -cvm -f tape0 -b 10k >brulogfile
```

DIAGNOSTICS

Most diagnostics are reasonably informative. The most common have to do with meaningless combinations of options, incompatible options, hitting memory or device limits, unresolved file links, trying to archive or restore something to which access is normally denied, or problems with media errors and/or archive corruption.

DEVICE TABLE

bru contains an internal table of known devices and their characteristics. bru first looks for an environment variable BRUTAB, which contains the name of the dynamically loaded file if it begins with a '/' character, or contains device descriptions if the first character is not '/'. If there is no BRUTAB environment variable, the file /etc/brutab is loaded. If neither of the preceding is found, an internal default description is loaded.

SIGNAL HANDLING

bru normally catches both interrupt (SIGINT) and quit (SIGQUIT) signals. When an interrupt is caught during archive creation or extraction, bru completes its work on the current file before cleaning up and exiting. This is the normal way of aborting bru. When a quit signal is caught, an immediate exit is taken.

Note that during file extraction, a quit signal may leave the last file only partially extracted. Similarly, a quit signal during archive creation may leave the archive truncated.

When either interrupt or quit is caught at any time other than during archive creation or extraction, an immediate exit is taken.

ERROR RECOVERY

When properly configured for a given software/hardware environment, bru can recover from most common errors. For example, attempts to use unformatted media are detected, allowing substitution of formatted media. Random blocks in an archive can be deliberately overwritten (corrupted) without affecting bru's ability to recover data from the rest of the archive. When I/O errors are detected, retries are performed automatically. Out of order sequencing on multi-volume archive reads is detected, allowing replacement with the correct volume.

DIRECTORIES

bru takes two actions with respect to directories that make creation and extraction of entire hierarchies of files more convenient and complete. These actions are automatic archiving of parent directories and automatic expansion of explicitly named directories.

Automatic archiving of parent directories means that when bru is given the complete pathname of a file to archive, it attempts to automatically archive all parent directory nodes necessary to fully restore the specified file. During extraction, any required directories which do not already exist are restored from the archive if possible, otherwise they are created with appropriate defaults for the current user. When bru reads it's list of files from the standard input, or when the -Pp option is given, this automatic archiving of parent directory nodes is suppressed. Note also that when creating archives with additional constraints on the selected files (such as use of the -n option), these parent directories may be excluded.

Automatic expansion of explicitly named directories means that when bru is given an explicit file name that names a directory node, not only is that directory node archived, but all files and subdirectories in that directory are archived. I.e., the entire file hierarchy rooted in the explicitly named directory is archived. When bru reads its list of files from the standard input, or when the -Pe option is given, this automatic expansion of directories is suppressed.

Note that incremental archives, archives created with the -Pp option, or archives created from a list of files supplied on the standard input stream, may not contain all of the necessary parent directories to replicate the original hierarchy and thus may result in creation of directories with the default attributes when files are extracted from the archive.

When bru reads the list of files from the standard input stream, the default values for the -P options are -PeFp, which turns off expansion of directories, turns on filter mode, and turns off automatic archiving of parent directories. This allows bru to be conveniently used to archive only filesystem nodes that are explicitly named on the input list.

When files are explicitly named on the command line (or default to '.'), the default values for the -P options are -PEfP, which turns on expansion of directories, turns off filter mode, and turns on automatic archiving of parent directories. This is typically the most convenient behavior for arguments given on the command line.

WILDCARDS

When reading archives, bru recognizes file name generation patterns in the same format as the shell. This allows greater flexibility in specifying files to be extracted, compared, or listed. As a special extension of shell-type expansion, the sense of the match is reversed for patterns that begin with '!'.

Note that the patterns may have to be quoted to prevent expansion by the shell. Also note that patterns are processed independently, without regard to any other patterns that may or may not be present. In particular, "/bin/a* /bin/b*" is equivalent to "/bin/[ab]*", but "/bin/!a* /bin/!b*" is equivalent to "/bin/*", not "/bin/![ab]*".

BYTE/WORD SWAPPING

While reading archives produced on other machines, bru automatically attempts to perform byte and/or word swapping as necessary.

REMOTE TAPE DRIVES

On 4.2 BSD systems, and System V systems that support networking, bru allows the use of remote tape drives for the archive device (via the -f option). A remote tape drive file name has the form

system[.user]:/dev/???

where system is the remote system, the optional user is the login name to use on the remote system if

different from the current user's login name, and /dev/??? is the tape drive to use (1600 BPI or 800 BPI, raw or blocked, rewinding or non-rewinding, etc.). In all cases, the user must have the appropriate permissions on the remote system. (See also the CAVEATS section, below.)

RUNNING FROM CRON

Sometimes it is convenient to run bru under conditions where there is no controlling terminal. This can be a problem if interaction is needed, such as when switching to a new volume. As an example, consider the case of running bru from cron, where the operator mounts a tape before leaving in the evening, and bru writes the first volume in the middle of the night. When returning in the morning, the operator wants to be able to mount a second tape if necessary, and instruct bru to continue.

If no interaction with the user is required, running from cron is no different than running directly from a terminal. However, when interaction is necessary there are basically two options; terminate, or find some way to communicate with the operator (or another program masquerading as the operator). The -B option provides for simple termination. The -I options provide for communication with an operator.

On systems that support fifos, a pair of fifos are used to send requests and receive replies. Before running bru, verifiy that these fifos are present if they are not then create the pair of fifos with the commands:

mknod /bru/bru.q p

mknod /bru/bru.r p

Then, add the arguments "Ib" or if you created any other name for your fifos then run the following command "-Iq,/bru/bru.q -Ir,/bru/bru.r" to the desired bru command line which ultimately gets executed undercron. The first time bru needs to communicate with an operator, it will open the two fifos, write a query to the bru.q fifo, and wait for a response from the bru.r fifo. A simple program provided with bru, called brutalk can be used to read the query and send a reply:

brutalk

brutalk /bru/bru.r

The brutalk program will continue to read queries and send replies until either bru exits, or a control-D (EOF) is typed at the terminal.

EXIT CODES

Bru always returns meaningful status as follows:

   0   Normal exit, no errors or warnings.

   1   Warnings (or interrupted).

   2   Errors (or quit received).

Appendix E – Table of BRU Modes and Options

Overview

The following table is a list of the bru modes. See Appendix C, "The BRU Manual Page," for Full descriptions.

```
bru mode [control options] [selection options] [files]
MODE
  -c     create a new archive with specified files
  -d     find differences between archived files and
    current files
  -e     estimate media requirements for create mode
    (media size must be known)
  -g     print only information from archive header
  -gg    print only the list of files created during
    the -G option (for pre-17.0 archives)
  -h     print help information
  -i     inspect archive for consistency and data
    integrity
  -t     list archive table of contents for files
  -x     extract named files from archive
```

The following table shows which of the BRU options are valid for each mode. See Appendix C, "The BRU Manual Page," for Full descriptions.

```
MODES          CONTROL OPTIONS
c,d,e,g,gg,h,i,t,x

*,*, , , ,  , , , ,    -a Reset file access times after
              reads.
*,*,*, , ,  , , ,*,* -b bufsize Set archive buffer size
              to bufsize bytes (scalable).
*,*,*, , ,  , , , ,    -B Background mode. No interaction
              with operator
 , , , ,  , , , ,* -C Always chown extracted files to
              the user's uid/gid
*,*, , , ,  , , , ,* -D On some systems, provides
              speedup via double buffering
*, , , , ,  , , , ,* -F No checksum computations or
              checking (dangerous for backup)
*,*,*,*, *,*,*,*,* -Ib Reads and writes to default
              fifos.
*,*,*,*, *,*,*,*,* -Il,pathname Write verbosity
              information to pathname
*,*,*,*, *,*,*,*,* -Iq,fifo Write interaction queries
              to fifo
*,*,*,*, *,*,*,*,* -Ir,fifo Read interaction replies
              from fifo
*,*,*, , ,  , ,*, ,* -j Enables a progress output for
              files larger than 500K
*, , , , ,  , , , ,    -l Suppress warnings about
              unresolved links
*, , , , ,  , , , ,* -L string In Create (backup) mode,
              BRU Labels the    archive with a given
              string from either the given string    or
              from a file (63 character maximum). In
              Extract (restore) mode, BRU is forced to
              check the archive label    and compare it
              to the given label. Can be disabled with
              -QL option.
*, , , , ,  , , , ,    -m Limit directory expansions to
              the same mounted   filesystem
*, , , , ,  , , , ,    -N level Use level for compression
              (default is 3). See -   Z
MODES          CONTROL OPTIONS
c,d,e,g,gg,h,i,t,x

*, , , , ,  , , , ,    -O Overwrite archive regardless of
              OVERWRITEPROTECT and    RECYCLEDAYS settings
```

```
 ,*, , ,   , ,*,*,* -p Pass over archive files by
                reading rather than     seeking.
*, ,*, ,   , ,*,*,* -Pa Turn off absolute to relative
                pathname conversion.
*, ,*, ,   , ,*,*,* -PA Turn on absolute to relative
                pathname conversion
*, ,*, ,   , , , ,   -Pe Turn off expansion of
                directories.
*, ,*, ,   , , , ,   -PE Turn on expansion of
                directories.
*, ,*, ,   , , , ,   -Pf Turn off filter mode (build
                internal file tree). For     stdin input
                only.
*, ,*, ,   , , , ,   -PF Turn on filter mode (do not
                build internal tree).
*, ,*, ,   , , , ,   -Pp Turn off automatic archiving of
                parent directory   nodes
*, ,*, ,   , , , ,   -PP Turn on automatic archiving of
                parent directory   nodes
*, , , ,   , , , ,   -QH Turns off placing small files
                into file headers (NOFILESINHDRS=YES)
 , , , ,   , , , ,* -QR Disable SmartRestore
 , , , ,   , , , ,* -QS Forces the translation of
                symbolic links during the    -T translate on
                restore option
*,*, , ,   , ,*,*,* -QV Ignore the incorrect volume
                warning in a multi-volume    operation
*, , , ,   , , , ,* -r rawdev Enable backup or restore
                of a raw data      partition
*, , , ,   , , , ,   -R Exclude remotely mounted
                filesystems (NFS/RFS)
```

```
MODES          CONTROL OPTIONS
c,d,e,g,gg,h,i,t,x
*,*,*, ,   , ,*,*,* -s size Specify the size of the
                archive media (K -       kilobytes, M -
                Megabytes, G - Gigabytes). This applies
                to files, tapes, diskettes, etc.
*, , , ,   , , , ,* -S size Turn on options to handle
                sparse files  intelligently and set
                threshold to size.
 , , , ,   , , , ,* -T filename Translates expression
                to new_expression as    defined in filename
*,*,*,*,   , , , ,   -U level Do not resolve directory
                levels greater than     level below the
                current directory                 level
*,*,*,*, *,*,*,*,* -v Enable verbose mode (-vv, -vvv,
                -vvvv available for    more verbosity)
*,*,*,*, *,*,*,*,* -V Print execution summary
*, , , ,   , , , ,* -w Prompt for action and wait for
                operator response
*, ,*, ,   , , , ,   -Z Enable software compression
                (also see -N)
```

Appendix F - Sample BRUTAB *File*

Overview

When completed, a typical brutab file will look like this. The first entry in the table represents the default archive device. If you do not specify a device on the command line (with the –f option), this is the archive device BRU will try to use.

```
#
#  *** GLOBAL BRUTAB PARAMETERS ****
#
#+OVERWRITEPROTECT=YES
#+ZBUFSIZE=500K
#+RECYCLEDAYS=14
#+MAXWRITES=200
#
#
#  FILE
#
#      brutab     bru data file for loadable device table
#
#
#  DESCRIPTION
#
#      Bru data file for loadable device table.
#      Note that the table MUST contain at least one entry
#      and the first entry is the default archive device.
#
#      Also, the table should contain an entry for the
#      standard input and output, with a name of "-".  This
#      entry SHOULD NOT be the first entry (or archives may
#      be inadvertently written to the user's terminal).
#
#      Entries consist of a device name field, followed by
#      one or more capabilities fields.  Entries may span
#      more than one line by escaping the newline at the end
#      of the line with a '\' character ('\' is last
#      character on line before newline).
#      All whitespace (tabs, blanks, newlines, formfeeds)
#      between fields is ignored.
#
#      The device name field must be the first field in the
#      entry and has the following form:
#
#              || ... |
#
#              ex:     /dev/rmt0|/dev/rmt1|/dev/rmt2
```

```
#
#      where each device has the same capabilities as the
#      other devices specified (a class of devices).
#
#      Each capability field is of the form:
#
#              =        or
#
#              ex:     size=640M   reopen  pwerr=EIO
#
#      Note that there can be no whitespace between the
#      capability name and the value.  Numeric values may be
#      given in absolute form or with a trailing scale factor
#      of:
#
#              b or B          Blocks (512 bytes)
#              k or K          Kilobytes (1024 bytes)
#              m or M          Megabytes (1024 * 1024 bytes)
#              g or G          Gigabytes (1024*1024*1024 bytes)
#
#      Currently used capabilities include:
#
# Name  Type            Meaning
# ————  ————            ——————————
# bufsize     numeric         default I/O buffer size for this
#                             device (32Kb if omitted)
#                             (beware of shared mem limits)
# size  numeric         media size in bytes if known,
#                             zero if unknown or variable.
# seek  numeric         minimum seek resolution,
#                             zero if no seeks allowed
# reopen      boolean         close and reopen archive upon
#                             media switch
# noreopen    boolean         no close and reopen archive upon
#                             media switch
# tape boolean         archive device is a tape drive
# rawtape     boolean         archive device is a "raw" tape
#                             drive
# rawfloppy   boolean         archive device is a "raw" floppy
# norewind    boolean         closing does not automatically
#                             rewind
#                             ("size" parameter should be
#                             zero)
# advance     boolean         read/writes advance media even
#                             when
#                             errors occur (most 9-track tape
#                             drives, few cartridge drives)
#                             WARNING: do not set if not true!
# qfwrite     boolean         query for OK on first write to
#                             device
# format      boolean         format media if necessary
# ignoreclose boolean ignore errors from closing the
#                             archive
#                             device (bugs in some drivers)
#
#############################################################
```

```
# Standard SCSI tape
# We set size=0 to allow the system to inform us when we
# reach the early warning (ENOSPC).
##############################################################
tape0 \
        size=0 seek=0 bufsize=32k shmseg=7 shmmax=200K \
        reopen rawtape tape rewind shmcopy


##############################################################
#  Entry for "norewind" tape - notice that noautoscan and
#  norewind flags are set
##############################################################
ntape0 \
        size=0 seek=0 bufsize=32k shmseg=7 shmmax=200K \
        reopen rawtape tape norewind noautoscan shmcopy \
        fmtcmd="tapectl -f ntape0 erase" \
        rfmcmd="tapectl -f ntape0 fsf" \
        bfmcmd="tapectl -f ntape0 bsf" \
        retencmd="tapectl -f ntape0 reten" \
        rewindcmd="tapectl -f ntape0 rewind" \
        eodcmd="tapectl -f ntape0 seod"


##############################################################
#  3.5" 1.44M floppy drives
##############################################################
/dev/fd0|/dev/fd0u1440 \
        size=1440K seek=2K bufsize=2k \
        format fmtcmd="fdformat /dev/fd0u1440" reopen shmcopy


#============================================================
#  This entry is a generic entry for stdin and stdout.  DO
#  NOT put this first or user's will probably inadvertently
#  write archives to their terminals.
#============================================================
- size=0 seek=0 bufsize=4k
```

Appendix G - BRU Execution Log

Overview

Every time BRU is executed, it writes a message to an execution log. The default log file is

```
/Library/Application Support/ArGest Backup/log/
bruexeclog.
```

Each entry in the execution log is a single line that contains the following information:

- date and time
- process ID of the BRU task
- ID of the user who started BRU
- message number
- message text

A line is written to the execution log when BRU is started and when BRU terminates. A line is also written to the log for each error or warning message. The execution log is designed to help you monitor your backups. Since it maintains a record of every time BRU was executed, you can tell how long it took to do a backup. Since it records all warnings and error messages, it is also useful for trouble-shooting. If a problem occurs while using BRU, the log will contain a record of what happened.

You can change the location of the execution log by setting the `BRUEXECLOG` environment variable. You should do this only if the default location is not satisfactory (because of limited space in that filesystem, for example).

The following two command lines entered in your currently active Terminal will temporarily change the location of the execution log to **/etc/newbrulog**.

```
BRUEXECLOG=/etc/newbrulog
export BRUEXECLOG
```

The log file should not take up much disk space, since most users do not execute BRU more than once or twice a day. Even a year's worth of log entries should take less than 100K bytes of space. We do not recommend it, but you can disable the execution log by setting `BRUEXECLOG` to a null string (i.e., `BRUEXECLOG=""`).

NOTE: If you have created a root and a boot floppy for crash recovery, you may need to set `BRUEXECLOG=/dev/null`. BRU will attempt to write to or create a `BRUEXECLOG` file whenever you run any of the BRU commands. This can cause you a problem if you do not have room on the boot device to create and append to the file.

Execution logging will also be disabled if BRU is renamed (or a link is created) and executed as `Bru` (with a capital B).

IMPORTANT NOTE: If you are having difficulties with BRU, please save the information in the execution log file. This will help BRU Technical Support to identify and solve your specific problem.

BRU Execution Summary

If BRU's verbosity is specified as `-vvvv` or greater, an execution summary will be printed when BRU completes execution. If you wish to print the execution summary only, without listing all the files, the `-v` (capital V) option should be specified.

Here's a typical execution summary:

```
**** bru: execution summary ****
Started:                Mon Feb 27 12:59:05 1995
Completed:              Mon Feb 27 13:01:38 1995
Archive id:             2f522f092adb
Messages:               0 warnings,  0 errors
Archive I/O:            3690 blocks (7380Kb) written
Archive I/O:            3690 blocks (7380Kb) read
Files written:          20 files (20 regular, 0 other)
Files read:             20 files (20 regular, 0 other)
Files in headers:       2
Write errors:           0 soft,  0 hard
Read errors:            0 soft,  0 hard
Checksum errors:        0
Min Compression:        41%
Avg Compression:        71%
Max Compression:        76%
Max Compressed Size:    2424613 bytes
Compression Overflows:  2
Difference count:       0
```

The fields are defined below:

- Started: Date and time the process started.
- Completed: Date and Time the process completed.
- Archive id: A unique 12-character archive identifier.
- Messages: A count of the total number of warning and error messages.
- Archive I/O: Number of BRU blocks (2Kb) written to the archive device. This should be zero if you are just reading (extracting files or inspecting) a tape
- Archive I/O: Number of BRU blocks (2Kb) read from the archive device. This should be zero if you are only writing to a tape (with no AUTOSCAN or inspection).
- Files written: Number and type of files written to the archive. This should be zero if you are just reading a tape.
- Files read: Number and type of files read from the archive device. This will be zero if you are just writing to a tape.
- Files in headers: Number of small files stored in header.
- Write errors: Number of errors counted when writing to the archive device. A "soft" error indicates a problem detected by BRU. A "hard" error is a hardware error reported by UNIX that caused BRU to issue an error message.
- Read errors: Number of errors counted when reading from the archive device.

- Checksum errors: Total number of checksum errors detected when BRU was reading the archive (in AUTOSCAN, extract or inspection modes).
- Min Compression: Minimum percent of compression for a file
- Avg Compression: The average compression of all files in the current archive.
- Max Compression: Maximum compression percentage.
- Max Compressed Size: largest file compressed.
- Compression Overflows: Number of files that were too large for the compression buffer (ZBUFSIZE).
- Difference count: Total number of files that were different when the -d option was used.

Appendix H - The Shell Scripts: fullbru *and* incbru

Overview

Two commands, `fullbru` and `incbru`, allow you to do full and incremental backups. These commands are shell scripts that can be run from the command line, or from `cron` for unattended backups (see Chapter 9, "Advanced Uses," "Running BRU from `cron`"). The standard `fullbru` command will back up all the files on your system. `incbru` will back up all the files that have changed since the last time `fullbru` was run. `fullbru` actually calls the `incbru` command with an argument of `0` (zero). The `incbru` script does all the work. The `incbru` script will perform incremental backups at levels from 0 through 9. The backup level is determined by an optional argument (a digit from 0 through 9) included on the `incbru` command line. If no argument is given, a level 1 backup will be performed. A level 1 backup will contain all files modified since the last full backup. A level 2 backup will contain all files modified since the last level 1 backup, and so on.

Here's how to do an incremental backup. Either command will back up all the files that have changed since the last full backup was run:

        # incbru

or

        # incbru 1

The following is a copy of the default `incbru` script. It was written to be as simple as possible and may not work correctly on your system. Feel free to modify it to suit your needs.

```
# This script will perform different levels of
# incremental backups using bru.
#
# It will back up all files that have changed
# since doing a lower level backup.
#
# If no argument is specified, the script will do a
```

```
# level 1 backup and will back up all files since
# the last full (level 0) backup. If the
# level 0 backup does not exist, it will do
# a level 0 backup instead.
#
# Feel free to customize this script to handle
# your specific backup requirements. This script
# is designed to do basic backups of the entire
# system from the root directory.
#
BRUDEV=""  # name of backup device. use default
#            if none is specified
BRUOPTS="-vvvv" # bru options
DIRS="./"  # directories or filesystems to
#            back up
LOGFILE="" # log file
MAILTO=root # where to mail the results
case "$#" in
  0)
    LVL=1  # no argument, so do a level 1 backup
    ;;
  1)
    LVL=$1
    ;;
  *)
    echo "usage: incbru [level]"
    exit 1
    ;;
esac
#
# check for legal backup level
#
case $LVL in
  0|1|2|3|4|5|6|7|8|9)
    ;;
  *)
    echo "incbru: $LVL is an illegal backup level"
    exit 1
    ;;
esac
#
# if level 0, do a full backup
#
DATEFILE=""
if [ "$LVL" -eq 0 ]
then
  DATEFILE=""
else
  PREVLVL=`expr $LVL - 1`  # previous backup level
  echo "PREVLVL = $PREVLVL"
  #
  # find date marker file for previous level backup
  #
  i=0
  while [ "$i" -le "$PREVLVL" ]
  do
    FILE=/etc/BRULEVEL$i
    if [ -f "$FILE" ]
    then
      DATEFILE=$FILE
      if [ "/etc/BRULEVEL${PREVLVL}" = "$FILE" ]
      then
        break # found previous date marker file
```

```
      fi
    fi
    i=`expr $i 1`
  done
fi
if [ "$DATEFILE" = "" ]
then
  if [ "$LVL" -ne 0 ]
  then
    echo "incbru: No previous backup less than Level $LVL"
    LVL=0
  fi
  echo "Performing Full Backup (Level 0)"
else
  echo "Performing Level $LVL Backup"
  BRUOPTS="$BRUOPTS -n $DATEFILE"
fi
#
# set up bru command that will do the backup.
# use default device from brutab if none is
# specified
#
if [ "$BRUDEV" != "" ]
then
  BRUCMD="bru -c $BRUOPTS -f $BRUDEV $DIRS"
else
  BRUCMD="bru -c $BRUOPTS $DIRS"  # use default
fi                                          # device
#
# create a temporary date marker file. if the
# backup is successful, this will be renamed
#
MARKER=/etc/BRULEVEL${LVL}
TMP=$MARKER.err
TS=`date`
echo "$TS - Started BRU Level $LVL Backup\n\tcommand = '$BRUCMD'" > $TMP
#
# change to the root directory and execute the
# bru backup. write output to log file if one
# is specified
#
BRUEXIT=2
cd /
if [ "$LOGFILE" != "" ]
then
  $BRUCMD > $LOGFILE 2>&1
  BRUEXIT=$?
else
  $BRUCMD 2>&1
  BRUEXIT=$?
fi
#
# check the exit code. if greater than or equal
# to 2, we've got a problem
#
TS=`date`
if [ "$BRUEXIT" -ge 2 ]
then
  echo "incbru: ERROR in BRU Backup"
  echo "$TS - ERROR IN BRU BACKUP, check log file: \
    '$LOGFILE'" >> $TMP
  mail $MAILTO < $TMP
  exit 2
else
  echo "BRU Level $LVL Backup successful"
  mv $TMP $MARKER
  echo "$TS - BRU Level $LVL Backup successful, \
    results in log   file: '$LOGFILE'" | mail $MAILTO
  exit 0
fi
```

Appendix I The brutalk *Program*

Overview

The following is a C language source code listing of the **brutalk** program. **brutalk** is a simple program that talks to a pair of fifos associated with a run of BRU. It uses the **–I** option and may be run under the control of the **cron** program.

NOTE: A compiled version of this program is included in your **/bru** directory.

```
/*****************************************************

brutalk.c simple program to talk to a pair of fifos for bru
DESCRIPTION
Simple-minded program to talk to a pair of fifos. Use as:
        brutalk [-t tty] /bru/bru.r
or
        brutalk [-t tty] /bru/bru.q /bru/bru.r

        Normally brutalk will attempt to communicate with the user
via /dev/tty. The -t option can be used to select any tty type stream
to open in place of /dev/tty. We fork to create two processes. The
child reads from the fifo where queries are posted and writes to the
terminal. The parent reads the replies from the terminal and writes
to the fifo where replies are expected. There are basically three
normal ways to exit:
        Before attempting to open explicitly named input and output
fifos, we set a timer. If the timer goes off because bru has not yet
opened the other end of the fifos, then we get a SIGALRM and exit.
This basically means that either bru has exited normally, or has not
yet requested any interaction.
        Note that the timeout is only effective until bru opens the
fifos for the first time. If you answer one query using brutalk, then
interrupt brutalk, and come back and run it again later, it will wait
until the next query is available. The initial timeout can be avoided
by using the redirection form, since it is actually the shell that is
opening the fifos in that case. The child dies when the program
posting queries exits, closing the write side of the query fifo. The
parent gets notified that the child has died by a SIGCLD and itself
exits.
        If the user types a ^D (EOF) then the parent sends the child
a kill signal and exits. The original client at the other end of the
fifos continues to execute. If it knows how to deal gracefully with
the fact that we have exited (by closing and reopening the fifos),
then we can reattach at a later time to answer more queries.
        This program is normally used to interact with bru when bru
is executed from cron by a command that includes the arguments:
        -I q,/bru/bru.q (send queries to fifo bru.q)
        -I r,/bru/bru.r (read replies from fifo bru.r)
        -I l,/tmp/brulog (send verbosity info here)
        The main function returns 1 to satisfy lint and also to cover
the "impossible case" where exit returns rather than exiting.
AUTHOR
Fred Fish
Tim Jones

        NOTE: This code sample is intended as an example ONLY.  It
will NOT compile as it exists.  The finished version of brutalk is
included with your BRU distribution and can be found in the /bru
directory.
*/


#include <stdio.h>
#include <signal.h>
#include <fcntl.h>
#include <sys/types.h>
#ifndef SIGCLD
#define SIGCLD SIGCHLD /* If no SIGCLD or SIGCHLD, we croak */
#endif
#if HAVE_PID_T
  typedef pid_t PID_T;
#else
  typedef int PID_T;
#endif
extern void perror ();
extern void exit ();
extern unsigned alarm ();
static void openfiles ();
static void passdata ();
static int rfifo;
static int wfifo;
static int ttyf;
int main (argc, argv)
int argc;
char *argv[];
{
  int optchar;
  char *infifo = NULL;
  char *outfifo = NULL;
  char *tty = NULL;
  extern int optind;
  extern char *optarg;
  while ((optchar = getopt (argc, argv, "t:")) != EOF) {
    switch (optchar) {
      case 't':
        tty = optarg;
        break;
      case '?':
        exit (1);
        break;
    }
  }
  for ( ; optind < argc; optind) {
```

```
    if (infifo == NULL) {
      infifo = argv[optind];
    } else if (outfifo == NULL) {
      outfifo = argv[optind];
    } else if (tty == NULL) {
      tty = argv[optind];
    } else {
      fprintf (stderr, "brutalk: too many arguments\n");
      fprintf (stderr,
        "usage: brutalk readfifo writefifo\n");
      exit (1);
    }
  }
  if (tty == NULL) {
    tty = "/dev/tty";
  }

  (void) signal (SIGALRM, exit);
  (void) alarm (5);
  openfiles (infifo, outfifo, tty);
  (void) alarm (0);
  passdata ();
  exit (0);
  return (1);
}

/*
Open the read and write fifos, and the terminal stream. Any failure
is fatal with an error message printed.
*/

static void openfiles (rname, wname, ttyname)
char *rname;
char *wname;
char *ttyname;
{
  if (rname == NULL) {
    rfifo = 0;
  } else if ((rfifo = open (rname, O_RDONLY)) == -1) {
    fprintf (stderr, "brutalk: can't open '%s'", rname);
    perror ("");
    exit (1);
  }
  if (wname == NULL) {
    wfifo = 1;
  } else if ((wfifo = open (wname, O_WRONLY)) == -1) {
    fprintf (stderr, "brutalk: can't open '%s'", wname);
    perror ("");
    exit (1);
  }
  if ((ttyf = open (ttyname, O_RDWR)) == -1) {
    fprintf (stderr, "brutalk: can't open '%s'", ttyname);
    perror ("");
    exit (1);
  }
}
```

```
/*
Fork to create bidirectional data paths and then loop, passing data
through each path until a termination condition occurs.
*/

static void passdata ()
{
  int inbytes;
  int outbytes;
  int pid;
  char inbuf[1024];
  char outbuf[1024];
  if ((pid = fork ()) == -1) {
    perror ("brutalk: can't fork");
    exit (1);
  } else if (pid == 0) {
    while ((inbytes = read (rfifo, inbuf, sizeof(inbuf)))
      > 0) {
      (void) write (ttyf, inbuf, (unsigned) inbytes);
    }
  } else {
    (void) signal (SIGCLD, exit);
    while ((outbytes = read (ttyf, outbuf,
      sizeof (outbuf))) > 0) {
      (void) write (wfifo, outbuf, (unsigned) outbytes);
    }
    (void) kill ((PID_T) pid, SIGTERM);
  }
}
```

Appendix J – The bruxpat _File_

The bruxpat File

```
# This file is used by the -X option to provide
# an inclusion/exclusion list. For each
# pathname of a file selected for backup,
# each line of this file is examined for a
# pattern and that pattern is applied to the
# pathname. If the pattern matches, the
# appropriate action is taken (the pathname
# is accepted or rejected). If the pathname
# makes it through all the patterns, it is
# accepted.
#
# Each command line in the bruxpat file (the
# file you are now reading) consists of a
# control field and a pattern. The
# pattern is separated from the control
# field by whitespace. Control field
# characters are:
#
# i Include this pathname if pattern
#  matches. The pathname is accepted and
#  no further patterns are applied.
#
#      *** NOTE ***
# bru stops trying on the first pattern
#  match found and passes the filename.
#  Since it scans patterns in the order
#  listed, include patterns should
#  usually be listed before any exclude
#  patterns.
#
# x Exclude this pathname if pattern
#  matches. The pathname is rejected
# and no further patterns are applied.
#
# s The pattern is a shell-style wildcard
#  pattern except that '/' characters are
#  not treated as special characters.
#
# r The pattern is a regular expression
#   (the same as used by the "grep" command).
#
# l The pattern is a literal string.
#
#
```

```
# Include the C runtime startup file but
# exclude all other object files.
#
is */crt0.o
xs *.o
#
# Exclude all core files
#
xs */core
xs core
#
# Exclude all files and subdirectories in
# the temporary directories.
# Handle files specified with relative and
# absolute pathnames.
#
xs ./usr/tmp/*
xs /usr/tmp/*
xs ./tmp/*
xs /tmp/*
#
# the bruxpat file also tracks files that should
# be excluded from compression attempts (-Z).
# Files and patterns listed here are not
# compressed.
#
zs *.[zZ]
zs *.gz
zs *.arj
zs *.bz
zs *.bz2
zs *.mp3
zs *.gif
zs *.zip
```

Appendix K - Advanced BRUTAB *Settings*

Overview

This section of the user's guide explains the advanced settings of the brutab file. These settings are either numeric or boolean entries. **NOTE**: These are advanced settigs and should only be changed if you are completely competent in your devices specific responces. We do suggest that these settings remain un-changed in your BRUTAB file.

Name	Type	Meaning
advance	Boolean	reads/writes advance media even when errors occur (most 9-track tape drives, few cartridge drives) WARNING: do not set if not true!
ederr	numeric	errno for end of device reached NOTE: "ederr" should be different than other **err's
eject	boolean	eject media after use (Macintosh style)
format	boolean	format media if necessary
frerr	numeric	errno for read of unformatted media
fwerr	numeric	errno for write of unformatted media
maxbufsize	numeric	maximum I/O buffer size for this device. Is set to 0 if unspecified or no limit.
noautoscan	boolean	disables automatic scanning of the archive after creation of an archive **IF YOU SPECIFY THIS BOOLEAN YOU WILL DEFEAT BRU'S AUTOMATIC ERROR CHECKING OF YOUR ARCHIVE.
prerr	numeric	errno for partial reads
pwerr	numeric	errno for partial writes
shmall	numeric	limit on total amount of shm used NOTE: applies only if "-D", double buffering, is specified when invoking bru.
shmcopy	boolean	tells bru that the archive device driver cannot do I/O directly from shared memory. NOTE: applies only if "-D", double buffering, is specified when invoking bru.
shmmax	numeric	limit on size of each shared memory segments NOTE: applies only if "-D", double buffering, is Specified when invoking bru.
Shmseg	numeric	limit on number of shared memory segments NOTE: applies only if "-D", double buffering, is specified when invoking bru.
wperr	numeric	errno for write protected media
zrerr	numeric	errno for zero length reads
zwerr	numeric	errno for zero length writes.

Appendix L - Using MOUNTCMD and UNMOUNTCMD

MOUNTCMD & UNMOUNTCMD

MOUNTCMD & UNMOUNTCMD allow the user to specify external commands to handle devices. MOUNTCMD and UNMOUNTCMD definitions are placed in the brutab file or defined as environment variables. MOUNTCMD= is used to specify a command that will be called before BRU attempts to open a device for reading or writing. The UNMOUNTCMD= should be used to specify a command that will be called after BRU has finished reading/ writing.

BRU passes four arguments to the commands specified by MOUNTCMD and UNMOUNTCMD. These arguments are described in the table below. In most cases, the commands specified will be shell scripts. However, BRU does not care what language is used to create these executable files as long as they are set to accept and act on the arguments listed below. Bellow you will find a reprint of the mountcmd.sh and unmountcmd.sh scripts supplied with BRU. This script like all other scripts supplied with BRU are to be modified by the user to fit their system.

Argument	M/U	Description
1	Both	Device Name
2	Both	Volume Number
3	Both	BRU Mode
4	M	Media size in Kbytes
4	U	Kbytes written/read

Two example shell scripts follow - please understand that these are examples and should be used simply as templates for a functional media changing mechanism - these scripts do absolutely nothing as they appear here:

BRU's sample mountcmd.sh script:

```sh
#!/bin/sh
############################################################
#
#  mountcmd.sh sample script for MOUNTCMD
#
# If the MOUNTCMD variable is set, this command will be called
# BEFORE BRU attempts to read or write a tape
#
# Please modify this script to suit your own needs
#
#
############################################################

CMD=$0          # name of this command
DEV=$1          # 1st parameter is device name
VOL=$2          # 2nd parameter is expected volume number
MODE=$3         # 3rd parameter is BRU mode letter
MSIZE=$4                # 4th parameter is media size in Kbytes


case "$MODE" in
  c ) :
    # Replace the following lines with any commands that
    # should be called before BRU attempts to WRITE to a tape
    #
    echo "$CMD: volume $VOL on device $DEV (mode = $MODE)"
    #
    # Sample command to create a file containing label info
    #
    echo "VOLUME $VOL LABEL" > /tmp/labelfile

    RTN=$?              # set return code
    ;;
  [idtgx] ) :
    #
    # Replace the following lines with any commands that
    # should be called before BRU attempts to READ a tape
    #

    echo "$CMD: volume $VOL on device $DEV (mode = $MODE)"
    RTN=$?              # set return code
    ;;
esac

exit $RTN       # return 0 if successful
```

BRU's sample `unmountcmd.sh` script:

```sh
#!/bin/sh
###############################################################
#
#  unmountcmd.sh        sample script for UNMOUNTCMD
#
# If the UNMOUNTCMD variable is set, this command will be
# called AFTER BRU is done reading/writing a tape.
#
# The UNMOUNTCMD will not be called unless the MOUNTCMD was
# specified. It is NOT possible to call the UNMOUNTCMD only.
# In most cases, the UNMOUNTCMD is not needed, as all the tape
# handling, jukebox commands, etc. can be done by the
# MOUNTCMD.
#
# Please modify this script to suit your own needs.
#
###############################################################

CMD=$0           # name of this command
DEV=$1           # 1st parameter is device name
VOL=$2           # 2nd parameter is volume number
MODE=$3          # 3rd parameter is BRU mode letter
IOSIZE=$4        # 4th parameter is Kbytes written/read


case "$MODE" in
  c ) :
    #
    # Replace the following lines with any commands
    # that should be called after BRU is done WRITING a tape
    #

    echo "$CMD: volume $VOL on device $DEV (mode = $MODE)"
    RTN=$?           # set return code
    ;;
  [idtgx] ) :
    #
    # Replace the following lines with any commands  |
    # that should be called after BRU is done READING a tape
    #

    echo "$CMD: volume $VOL on device $DEV (mode = $MODE)"
    RTN=$?           # set return code
    ;;
esac

exit $RTN      # return 0 if successful
```

Appendix M - BRU Messages

Overview

All BRU messages are associated with a number. The message number consists of a single letter followed by three digits. The first letter of the message number indicates the type of message (i.e.: **[I181]** is an Informational Message). The messages are listed below:

A - alert message

BRU is alerting the user of a problem. This problem or alert, is not as strong as an error or a warning message but none the less BRU will require a response. This is message is usually followed by a "Q" message

E - error message

BRU has found that the current operation has errors that it cannot overcome. Depending on the **MAXERRORS** setting in your **BRUTAB**, BRU will terminate with an execution summary exit code of 2.

I - informational message

Information regarding the current process. This type of message is echoed to the **/var/log/bruexeclog** file as an "L" message.

L - log message

Entry message to the **/var/log/bruexeclog** regarding the current process. Entry is a copy of what is sent to the screen with the "I" message. All entries are made with a date and time stamp.

Q - query message

BRU is waiting for a response to complete the current process.

W - warning message

BRU has found problems with the current operation. The errors where not serious but BRU was unable to resolve the issue. Depending on your setting for **MAXWARNINGS** in your **BRUTAB** file BRU will terminate the current process.

NOTE: You should investigate all WARNING and ERROR messages reported by BRU. These messages can result in not being able to restore your data.

Many BRU messages are the result of errors returned by the UNIX system. For these types of errors, the UNIX error message is included as part of the BRU message. This portion of the message is described below:

    errno = code, description

    code is the UNIX error number.
    description is a brief text message associated with the error number. These are system values and are not BRU errors. BRU simply reports them as they are defined by the system.

The BRU messages are listed on the following pages. Following each message is an explanation and possible suggestions for corrective action.

BRU Message Code Listing

[E001] specify mode (-cdeghitx)

The user ran BRU without specifying the mode. The user must specify a valid mode. See Appendix D for a list of BRU modes.

[W002] filename: can't open archive: errno = code, description

BRU couldn't open the archive file filename for some reason. The reason is indicated by the UNIX error message. Make sure that the

device actually exists. Also see if another process (or an old BRU process) still has control of the device.

[W003] filename: warning - close error on archive: errno = code, description

BRU received an error from the operating system when BRU attempted to close the archive. The reason is indicated by the UNIX error message.

[W004] warning - archive read error at block blocknum: errno = code, description

BRU received an unrecoverable error when attempting to read an archive. The reason for the error is indicated by the UNIX error message. Whatever data was available at that location in the archive is unrecoverable.

[W005] warning - archive write error at block blocknum: errno = code, description

BRU received an unrecoverable write error while attempting to write an archive. The reason for the error is indicated by the UNIX error message. Whatever data that was to be written at that location in the archive has been discarded. Proper corrective action depends upon the situation and the specific file within which the error occurred. If this error occurred on the first block of an archive, the archive may be write protected, in which case, the wperr parameter is probably set incorrectly in the brutab file.

Another possibility is that the I/O buffer size is too large for the given device. Experiment with a smaller buffer size. Set the buffer size with the bufsize parameter in the brutab file or use the -b option to set it on the command line.

[E006] seek error on archive: errno = code, description

BRU received an unrecoverable seek error from the operating system on an archive file. The reason for the error is indicated by the UNIX error message. Make sure that your archive device actually has the capability to do random seeks to any location. Most tape drives do NOT have this capability. Check the value of the seek parameter in the brutab file. If your device does not have the ability to do random seeks, you must set seek=0.

[E007] media size smaller than I/O buffer size!

This error was caused when BRU detected a media size that was inconsistent with the I/O buffer size. The media size should be at least as large as the I/O buffer size. It is not possible to write to device if the media size smaller is smaller than the buffer size (i.e. BRU cannot write 1 megabyte of buffer data to a floppy with a media size of 720 kilobytes). Check the brutab file and make sure that the bufsize parameter is less than size.

[W008] warning - buffer size bufsize exceeds maximum maxbufsize allowed for device

Check the brutab file and make sure that bufsize is less than maxbufsize.

[E009] can't allocate bufsize archive buffer: errno = code, description

This message was caused when BRU could not allocate an I/O buffer of the size requested. Try reducing the buffer size to a smaller value. This can be done with the **-b** command line option, or the **bufsize** parameter entry in the **brutab** file.

[W010] filename: warning - block sequence error

BRU detected an inconsistency in the ordering of blocks returned by the archive device on a read.

For example, BRU asked for blocks "11, 12, 13, 14, 15" and got blocks "11, 12, 14, 15, 16." This error may indicate a hardware problem. If this message occurs at the start of the 2nd volume (when reading a multiple volume archive), it probably means that the last few blocks of the previous volume were missing or were unreadable. This can occur if the media size parameter was set too large in the brutab file (i.e. size=150M for a 60 megabyte tape) during the backup and BRU attempted to write past the end of the previous volume. With some tape drives, it may also occur if the media size was set to zero (unknown). The problem can usually be avoided by setting the media size to a value that is certain to be less than the maximum capacity of the volume (i.e. set size=149MT for a 150 megabyte tape).

To detect and avoid this problem, always verify your archive immediately after the backup is performed. **AUTOSCAN** does this automatically for all devices that have an entry in the brutab file (unless **noautoscan** is set or you are using a **norewind** device).

[W011] `warning - file synchronization error; attempting recovery ...`

BRU was expecting to find a file header block while reading an archive, but instead found another type of block. This warning will occur if you started reading an archive at a volume other than the first, or skipped a volume in the middle of reading an archive. This error message can be suppressed with the `-Qv` option flag. For more information on the optional flag see Chapter 8. BRU will scan each successive archive block looking for a file header block, and normal processing will resume from the first file header block found.

[W012] `filename: no file: errno = code, description`

The named file does not exist or part of the path name of the named file is not searchable given the current permission settings.

[E013] `filename: can't stat: errno = code, description`

The UNIX stat system call failed. This means that BRU was unable to obtain status information (ownership, access and modification times, link count, etc.) on the file. The reason for the error is indicated by the UNIX error message. Generally, this error is caused because the file is not accessible with the user's current permission settings. On networked systems, it may occur when trying to access remote (NFS-mounted) files. If this is the case, you should check your network permission settings.

[E014] `pathname path too big (1023 max)`

This error was caused when BRU detected a pathname longer than 1023 characters. No known UNIX system allows pathnames longer than 1023 characters, so this message may indicate that the filesystem is potentially corrupted or that something else is seriously wrong.

[E015] `*** OBSOLETE MESSAGE NUMBER ***`

[E016] `filename: can't open: errno = code, description`

BRU could not open filename. The reason is given as part of the UNIX error message. In many cases, this is caused by insufficient file access permissions.

[W017] `filename: warning - file close error: errno = code, description`

BRU received an error when attempting to close the file filename. The reason for the error is given by the UNIX error message.

[E018] `filename: read error: errno = code, description`

BRU received an error while reading a file. The reason is indicated by the UNIX error message. This message means that the file was not backed up properly. It may be an indication of hard disk failure, a corrupted filesystem, a damaged file that is unreadable, or other system problems (like an NFS problem.)

[W019] `filename: warning - file was truncated`

The file filename was truncated while BRU was in the process of reading or writing it. Usually, another program has modified the file that BRU was reading. This can occur if you are attempting to back up database files and the database program is active. If BRU was creating an archive at the time, the archived file is padded with sufficient null characters to bring it back to the size it was originally (the size of the specified file when BRU began to archive it). This is the same size recorded in the file header block. If this message occurs while backing up, the data archived from filename is probably not correct (because the data was changed while BRU was reading it). Even though BRU may restore this file later without any warnings, the data could contain errors. If this message occurs while restoring an archive, it indicates that a problem occurred and that BRU was unable to restore data from the last part of the file.

[W020] `filename: warning - file grew while archiving`

The file filename grew in length while BRU was in the process of reading it. If BRU was creating an archive at the time, the archived file was truncated to it's original size (the size of the file when BRU started to read it). This is the size recorded in the file header block. This warning is commonly seen for log or database files, to which information is constantly being added. It can generally be avoided by backing up the system in single user mode (or by shutting down the database before doing a backup). If the file causing this message is not critical (i.e. log files like `/var/log/bruexeclog`) you may wish to exclude these files from the backup. This can be done by specifying a pattern in the `bruxpat` file and using the `-QX` option.

[W021] `filename: warning - can't set user id: Not owner`

BRU attempted to extract filename which was stored with the `suid` bit set and the user running BRU was not the original owner of the file (and did not have superuser privileges).

[W022] `filename: warning - can't set group id: Permission denied`

BRU attempted to extract filename which was stored with the **sgid** bit set and the user running BRU was not the original owner of the file (and did not have superuser privileges).

[E023] *filename*: **can't exec: errno = code, description**

BRU could not execute the file filename for the reason given as part of the UNIX error message. Generally, this error occurs because filename does not exist, or it was not executable by the user running BRU.

[E024] can't fork, try again: errno = code, description

This error was caused when BRU couldn't execute a fork system call. The reason is indicated by the UNIX error message. Generally, this indicates the system is in serious trouble, or the per-user limit on processes has been exceeded.

[E025] unrecognized wait return statcode

The wait system call returned a status code of statcode which BRU was not able to understand. If this error occurs it may indicate that there is a problem with your system or that your version of BRU is incompatible with your current version of UNIX. If you have upgraded your operating system, you may also need to upgrade your copy of BRU.

[E026] child interrupted: errno = code, description

The child process which BRU was waiting for was interrupted. The reason for the error is indicated by the UNIX error message.

[E027] *filename*: **fatal error; stopped by signal sigcode**

The child process, filename, which BRU was waiting for was stopped by a UNIX signal. The reason is indicated by **sigcode**.

[E028] *filename*: **fatal error; terminated by signal sigcode**

The child process, filename, which BRU was waiting for was terminated by a UNIX signal. The reason is indicated by **sigcode**.

[W029] *filename* **core dumped**

BRU was waiting for a child process, filename, which terminated abnormally and dumped to core.

[E030] inconsistent wait status wait code

BRU received an unexpected return code of wait code from a UNIX wait system call. Usually, the child process which was being run has gone berserk in some manner. This error may indicate that there is a problem with your system or that your version of BRU is incompatible with your current version of UNIX. If you have upgraded your operating system, you may also need to upgrade your copy of BRU.

[E031] can't set uid to userid: errno = code, description

BRU received an error when trying to run the **setuid** system call. The reason is given by the UNIX error message. If this error occurs, it indicates a possible UNIX system bug or an internal bug in BRU.

[E032] can't set gid to groupid: errno = code, description

BRU received an error when trying to run the **setgid** system call. The reason is given by the UNIX error message. If this error occurs, it indicates a possible UNIX system bug or an internal bug in BRU.

[W033] *filename*: **warning - error count block checksum errors**

While reading an archive, BRU detected **errcount** number of checksum errors in the specified file. This message can occur if the archive was originally written with errors-possibly caused by a buffer size setting that is too large. Try setting the buffer size to a smaller value when creating the archive (i.e. set **bufsize=10K** in the **brutab** file or use the **-b 10K** option on the command line). When reading an archive, BRU normally attempts to read the archive with the write buffer size (it obtains the write buffer size from the archive header block stored at the start of the archive). The proper buffer size varies with the system and type of tape drive. In some cases, a tape written on one system (with a large buffer size like 64K) cannot be read properly on another system (which can only handle a small buffer size like 10K). If this is the case, you may be able to force BRU to read the tape by forcing the buffer size to smaller value (i.e. specify **-b 10K** as one of the command line options). Checksum errors may also be caused by hardware or tape problems. Try cleaning the heads on your tape drive. Try to retention the tape. Also, make sure that your tape cartridges are in good shape-tapes do not last forever. They should be rotated frequently and replaced on a regular basis. Make sure that you are using the proper kind of tape with your tape drive. Many tape cartridges look the same (especially 1/4" tapes), but have different densities. For instance, if you have a 150 MB tape drive, you should use 150 MB (DC6150) or 250 MB (DC6250) tape cartridges. You will get errors if you try to write to a 60 MB (DC600A) tape.

[E034] internal bug in routine routinename

BRU detected some sort of internal bug in the routine **routinename**. If this error occurs, it may possibly be a bug in BRU, or a hardware or

kernel software problem. If it is not repeatable, it is likely to be a hardware or kernel bug. This message should be reported to TOLIS Group Tech Support.

[E035] `can't allocate byte_count more bytes: errno = code,` `description`

BRU ran out of memory for some reason. This error generally occurs when BRU tries to create a tree from a list of filenames read from the standard input, or when memory is very limited on the system due to hardware or CPU constraints.

[E036] `internal error in tree; pathname overflow`

While building a file tree, BRU created a path which exceeded 1023 characters in length. No known UNIX system allows pathnames longer than 1023 characters, so this message may indicate that the filesystem is corrupted or that something else is seriously wrong.

[E037] `*** OBSOLETE MESSAGE NUMBER ***`

[E038] `filename: seek error: errno = code, description`

BRU received an error when attempting to seek to a certain location in the file that it was reading or writing. The name of the file is indicated by filename. The reason for the error is given by the UNIX error message. This error is rare and usually indicates a hardware problem with the disk drive.

[W039] `warning - info block checksum error`

BRU detected a checksum error while reading the first block of an archive. The info block (archive header block) contains information about the archive which is of use to BRU, but not critical to reading or extracting files from the archive. Make sure that the archive you are attempting to read was actually written by BRU. This error often occurs when attempting to read a tape written by another program (like `tar` or `cpio`). It may also occur if you try to read a tape that is blank. If this is the ONLY warning or error message (and BRU appeared to work normally), it can usually be ignored. In this particular case, it means that BRU had trouble reading the first block of the archive, but was able to skip past the first block and read the rest of the archive normally.

[E040] `filename: write error: errno = code, description`

BRU received an error when attempting to write to the file filename. The reason for the error is shown by the UNIX error message. If this error occurs, it usually indicates a hardware problem with your hard disk. It could also indicate that the filesystem containing filename is out of space, or that the filesystem is write- protected (it may be mounted as "read-only" or you may not have write permission).

[W041] `filename: warning - error setting mode: errno = code,` `description`

BRU received an error when attempting to set the mode of filename. The error occurred when BRU was executing the system call `chmod`. The reason is indicated by the UNIX error message.

[W042] `filename: warning - error setting owner/group: errno =` `code, description`

BRU received an error from the operating system when attempting to set the owner id or group id of filename. The error occurred when BRU was executing the system call chown. The reason is indicated by the UNIX error message. On systems which support symbolic links, this error can occur when BRU attempts to set the owner/group id of a symbolic link which points to a file which does not exist. This can occur if the symbolic link filename is restored, but the file the symbolic link points to is not restored.

[W043] `filename: warning - error setting times: errno = code,` `description`

BRU received an error from the operating system when attempting to set the access and modification times of filename. The error occurred when BRU was executing the system call `utime`. The reason is indicated by the UNIX error message.

[E044] `filename: error making node: errno = code, description`

BRU received an error when attempting to create a special file system node, such as a FIFO, block special file, or character special file. The error occurred when BRU was executing the system call `mknod`. The reason is indicated by the UNIX error message. This message may occur when trying to restore special files and you do not have superuser privileges. On some systems, only the root user has the ability to create special files.

[E045] `filename1: can't link to filename2: errno = code,` `description`

BRU received an error when attempting to make a hard link between `filename1` and `filename2`. The reason is indicated by the UNIX error message. This message occurs when `filename2` already exists and cannot be overwritten by a link.

[E046] internal error; inconsistent phys blk addrs

[W047] warning - missing archive header block; starting at volume volnum

BRU couldn't find an archive header block at volume number **volnum**. This warning is normal when BRU is asked to start reading an archive at some other volume than the first volume. For example, you will see this message if you immediately try to restore files from the 3rd tape of an archive (without reading through the 1st and 2nd tapes). If you want to start from a different tape other then the first tape. You must use the **-Qv** option, this will suppress the error message.

[W048] filename: warning - lost linkage: errno = code, description

BRU could not preserve the linkage of two files. The reason is indicated by the UNIX error message. Generally, this error is seen when BRU ran out of memory when it attempted to allocate memory internally to maintain the linkage information of the specified file. In this case, the file would be archived as two separate, distinct files in the archive. Only the linkage information would be lost.

[W049] filename: warning - linknum unresolved link(s)

While archiving the file filename, BRU detected **linknum** number of unresolved links to filename. This error is generated when there is still another pathname which points to filename which does not appear in the archive. Usually, this message occurs when BRU is asked to archive a set of directories that contain files that have hard links to files located in other directories (that are not archived by BRU). This message can be disabled by specifying the **-l** option on the command line.

[W050] ttyname: warning - can't open for interaction: errno = code, description

BRU could not open the **tty** stream **ttyname** to interact with the user. The reason for the error is given by UNIX error message. This message may occur when attempting to run BRU in the background and the **-B** option (background mode) has not been specified. When run in the foreground, BRU attempts to use the **/dev/tty** device to communicate with the user. In background mode, **/dev/tty** is not available. In this case, the interaction pathnames can be specified with the
-Iq,queryfile and **-Ir,replyfile** options on the BRU command line (this is normally used when running BRU with the **brutalk** program).

[E051] date conversion error: date

The string specified by date is not in the proper format or is not a legal date and time.

[W052] warning - uname failed: errno = code, description

BRU received an error when attempting to execute the **uname** system call. The reason is indicated by the UNIX error message.

[W053] warning - label string too big

BRU has a string length limit of 63 characters for a user specified label (used with the **-L** option). You must shorten the length of your label string.

[E054] error - invalid uid/filename as -o argument: pattern

BRU could not convert a given symbolic user name to the internal numeric form. This error usually occurs when the **-o** option is used and BRU cannot find username in the **/etc/passwd** file.

[E055] error - illegal wildcard pattern: pattern, errmsg

The wildcard matching pattern specified by pattern is not legal. The reason is indicated by **errmsg**.

[E056] *filename*: can't overwrite: errno = code, description

The file filename could not be overwritten during extraction. The reason is indicated by the UNIX error message. In most cases, this message is due to a permissions problem.

[W057] *filename*: can't access for write: errno = code, description

The file filename could not be accessed for write. The reason is indicated by the UNIX error message. In most cases, this message is due to a permissions problem.

[W058] *filename*: can't access for read: errno = code, description

The file filename could not be accessed for read. The reason is indicated by the UNIX error message. In most cases, this message is due to a permissions problem.

[W059] *filename*: warning - will not be contiguous: errno = code, description

BRU was unable to create the file filename as a contiguous file. The reason is indicated by the UNIX error message. This message should only occur on systems that support contiguous files (like **Masscomp** or **HP-UX**).

[W060] *filename*: **warning - contiguous files not supported, extracted as a regular file**

The file filename cannot be restored as a contiguous file, so BRU will create a regular UNIX file instead.

[E061] **can't read both file list and archive from stdin!**

BRU was instructed to read both an archive and a list of files from the standard input stream. This error occurs when an illegal BRU command like the following is entered: **bru -x -f - -**

[W062] **warning - premature end of volume volnum**

When reading/writing an archive device, BRU encountered an end-of-file (or got an I/O error) before reaching the expected end of the archive. This message is often preceded by messages [W004] or [W005]. In this case, it may indicate a problem with the tape drive hardware, old or damaged tapes, or incompatible tape formats (i.e. trying to write to a 60MB tape cartridge on a 150MB tape drive).

[W063] **warning - media appears to be unformatted: errno = code, description**

When BRU first attempted to read/write to a device it received an error. The reason is indicated by the UNIX error message. If BRU receives an error on the first read or write to an archive device, and the error conditions match the values set in the **brutab** entry for unformatted media in this device, BRU will issue this warning message. When writing, if the format and **fmtcmd=** parameters are set for the device, this warning will be suppressed and BRU will attempt to format the media.

[O064] ***** OBSOLETE MESSAGE NUMBER *****

[W065] **warning - using internal default device table**

BRU could not find the brutab file specified by the BRUTAB environment variable, or the default brutab file located in **/etc/brutab**. In this case, BRU used its internal brutab, which may not be correct for the current archive device.

[I066] *filename*: **not restored**

This is an informational message. BRU did not restore the file filename because the current file (on the disk) has a modification time that is newer than the file read from the archive. This is BRU's default method of restoring files. If you wish to overwrite all files, regardless of date, you should add the **-ua** option to the BRU command line.

[W067] **warning - media appears to be write protected or wrong density**

BRU received an error on its first attempt to write to an archive device. BRU has determined that it might be caused by media that is write-protected. The UNIX system may not return the proper error code, so it is not always possible for BRU to determine if the media is actually write-protected. BRU tries to determine the write-protect status by comparing the **errno** code returned by UNIX with the value of **wperr** (as specified for the device in the brutab file). If **wperr=0** (or is not set) then BRU must "guess" at whether the device is truly write-protected. In this case, it assumes that an error on the first write attempt is caused by write-protection, and issues the above message.

[W068] *filename*: **warning - not found or not selected**

The user specified a file on the command line which BRU did not find. The file filename may not exist or may be spelled incorrectly. If you are attempting to extract (restore) a file, make sure that filename EXACTLY MATCHES with the desired filename on the archive, including any beginning slashes. For example:

/myfile DOES NOT MATCH ./myfile.

[W069] **warning - may have to use -F option to read archive**

BRU encountered an archive which does not appear to have checksum. The archive may have been written with **-F** option (which is not recommended) and must be read with the same option. This message sometimes occurs when BRU attempts to read an archive that was written by another program, like **tar** or **cpio**. It can also occur when BRU has trouble reading a BRU archive due to bad tapes, dirty tape heads, hardware problems, incompatible tape formats, etc.

[E070] **interaction needed, aborted by -B option**

BRU was run with the **-B** option, indicating that it is running in background mode and that no user interaction is possible. It encountered a condition that required user interaction (like loading a new tape) and terminated. The **-B** option is normally set automatically

when BRU is started in the background, so this message may occur even if −**B** was not explicitly specified.

[E071] *filename*: **error making directory: errno = code, description**

BRU received an error when attempting to create a directory. The reason is indicated by the UNIX error message. In most cases, this occurs when the user has insufficient permissions.

[E072] *filename*: **error reading symbolic link: errno = code, description**

BRU could not read a symbolic link for some reason. The reason is indicated by the UNIX error message.

[E073] *filename*: **symbolic links not supported**

While running on a system that does not support symbolic links, BRU encountered a symbolic link while comparing an archive in differences mode (−d option).

[E074] *filename*: **could not make symbolic link: errno = code, description**

While extracting the symbolic link filename, BRU was unable to create a symbolic link. The reason is indicated by the UNIX error message. This error will occur if your version of UNIX does not support symbolic links.

[E075] *filename*: **could not make fifo**

BRU tried to extract a FIFO (named pipe file) on a system which does not support FIFOs. Normally, BRU tries to create a regular file with the same name. In this case, the attempt to create a regular file was unsuccessful.

[W076] warning - link of filename to dirname, dirname is a directory, no link made

BRU was asked to create a symbolic link from filename to the directory **dirname**, on a system which does not support symbolic links. Since hard links to directories are not allowed by UNIX, this warning is issued and no link is made.

[W077] warning - link of filename1 to filename2, filename2 does not exist

BRU attempted to create a hard link from **filename1** to **filename2** and **filename2** does not exist. Generally, this message occurs when BRU is asked to do a partial restore and **filename2** is not present.

[W078] warning - extracted fifo filename as a regular file

BRU was asked to extract a FIFO named filename on a system which does not support FIFOs. It extracted filename as a regular file. The correctness or desirability of this behavior is subject to debate, which is why the warning is issued.

[W079] *filename*: **warning - linkcount additional link(s) added while archiving**

While BRU was archiving a file, there were **linkcount** additional links made to it. These additional links may or may not have been archived.

[W080] *** OBSOLETE MESSAGE NUMBER ***

[E081] no default device in brutab file, use -f option

BRU could not find a default device in the **brutab** file. The default device is always the first device entry in the **brutab** file.

[EO82] *** OBSOLETE MESSAGE NUMBER ***

[W083] warning - attempt to change buffer size from oldbufsize to newbufsize ignored (incompatible brutab entries)

BRU detected different default buffer sizes when reading or writing to multiple devices (device cycling). The buffer size is not allowed to change between volumes of an archive. This error usually occurs at the start of the second device, when BRU reads the **bufsize** parameter for that device (from the **brutab** file) and discovers that the buffer size differs from the size used by the first device. To avoid this warning message, use the −**b** option to force a specific buffer size for all devices.

[E084] double buffering I/O error, bytecount bytes read/ written: errno = code, description

[E085] problem setting up double buffering, using normal buffering

Both of these errors indicate that BRU encountered a problem setting up the double buffering. Sometimes, reducing the I/O buffer size will remedy the problem.

[E086] *filename*`: media ejection failed: errno = code,`
`description`

> On systems which support ejection of archive media under software control, BRU may be configured to eject each media when it is done with the media. BRU encountered some sort of error while attempting to eject the media.

[I087] *filename*`: compressed version was larger, stored`
`uncompressed`

> When file compression is utilized via the `-Z` option, BRU will check to ensure that the compressed version of the file uses fewer archive blocks than the uncompressed version. If the compressed version will not result in any savings in archive space (it is larger than the normal file), then the uncompressed version will be archived instead.

[E088] *filename*`: decompression failed (errmsg)`

> BRU received an error from when attempting to decompress a file. The reason is indicated by `errmsg`. The file filename has not been extracted properly and may contain errors.

[W089] `warning - estimate mode ignores compression`

> BRU was told to use both the `-e` and `-Z` options simultaneously. Because of the large overhead in compressing files, and because there is no way to determine the compression ratio without actually doing the compression, BRU cannot estimate how much archive space is required for an archive when compression is enabled. Therefore, the `-e` option ignores possible savings due to compression.

[W090] *filename*`: warning - not deleted: errno = code,`
`description`

> BRU received some sort of error while attempting to delete (unlink) filename.

[W091] *filename*`: warning - compression failed, stored`
`uncompressed`

> BRU received an error from UNIX when BRU attempted to compress filename for storage (such as a filesystem temporary space overflow). BRU could not generate the compressed version of the file. Thus, the file was stored uncompressed.

[E092] *** OBSOLETE MESSAGE NUMBER ***

[W093] `warning - buffer size bufsize exceeds system imposed`
`limit buflimit with double buffering`

> While attempting to set up double buffering using System V Style shared memory support, BRU was asked to use an I/O buffer size which resulted in the double buffering buffers exceeding the system imposed shared memory limits. Try setting `shmmax` to a smaller value.

[W094] `warning - buffer size automatically adjusted to`
`bufsize`

> While attempting to set up double buffering using System V Style shared memory support, BRU was asked to use an I/O buffer size which resulted in the double buffering buffers exceeding the system imposed shared memory limits. The I/O buffer size was automatically adjusted downwards to the maximum size which the system could support.

[E095] `could not get shared memory segment: kilobytes:`
`errno = code, description`

> BRU was attempted a system call to `shmget` and was unable to get the requested amount of shared memory. The reason is indicated by the UNIX error message.

[E096] `could not attach shared memory segment: errno = code,`
`description`

> BRU made a system call to `shmget` which failed after BRU had already made a successful `shmget` call. The reason for the error is indicated by UNIX error message.

[E097] `could not allocate message queue: errno = code,`
`description`

> BRU could not allocate the memory needed to perform double buffering (`-D` option). The reason is indicated by the UNIX error message. Your system may not support shared memory, or the shared memory parameters (`shmmax`, `shmseg`, `shmall`) may not be set correctly in the `brutab` file.

[E098] `warning - don't understand -I option badargs`

> The string `badargs` was not recognized as a valid argument for the interaction option `-I`.

[W099] `warning - need more than segments shared memory`
`segments`

BRU was not able to allocate enough shared memory segments. Try setting the **shmseg** parameter for the device in use to a lower value.

[W100] warning - failed to move break value by {number of bytes} bytes: errno = code, description

BRU made a system call to **sbrk** which failed. BRU was unable to adjust the break value. This message is only caused when BRU was previously able to adjust the break value to the desired place; or should be able to adjust the break value, such as when reducing the amount of memory used.

[W101] warning - compression initialization failed, -Z suppressed

BRU could not acquire sufficient memory to perform the requested file compression. Compression was not performed. Try reducing the number of bits in compression by using the **-N** option with a lower value.

[W102] warning - unknown child died, pid pidnumX(expected pidnum), status statcode

While waiting for a specific child process to exit, the wait system call returned to BRU the pid of another process, **pidnumX**, which exited with the status of **statcode**. This error should never occur. If this error occurs, it is usually indicative of a serious problem with the system.

[E103] double buffer child died, status statcode

The child process used by BRU for double buffering died unexpectedly. The reason is indicated by the UNIX status code **statcode**.

[E104] warning - double buffer child error errcode

The child process used by BRU for double buffering received some sort of fatal error, which the child process was able to recognize as unrecoverable. The reason for the error is indicated by **errcode**.

[W105] warning - no double buffer child to reap: errno = code, description

BRU was waiting for a double buffer child to exit and the wait system call failed for some reason which was unexpected in the parent process.

[W106] warning - archive device may need "shmcopy" flag set in brutab entry

On some systems, the device driver for a given archive device may not be able to do I/O directly to or from shared memory. BRU detects this condition when the first write to, or the first read from, a given device fails with UNIX error code **errno** set to **EFAULT**. BRU issues this warning message and automatically attempts to switch to a mode where the data is copied to or from a local buffer. This automatic switching generally succeeds on writes and fails on reads, which is why the suggested fix is printed as a warning message.

[E107] *filename*: error - unrecoverable archive write error, some data lost: errno = code, description

BRU received an unrecoverable write error while creating an archive, and all or part of the data was lost for filename. This message may be an indication of tape hardware problems, dirty tape heads, an improper BRU buffer size, tapes that need retensioning, or tapes that are simply worn out. On high-density tape drives, this message can occur when an attempt is made to write to a low-density tape. Usually this happens with 1/4" tape cartridges, which all look similar. For example, this error will occur when using a 150MB tape drive to write to a DC600A (60MB) or DC300 (30MB) tape cartridge. High-density tape drives can normally read low-density cartridges, but they cannot write to them.

[W108] warning - media appears to be unformatted or write protected: errno = code, description

This is a general warning which may appear on the first attempt to read or write an archive volume which is unformatted, or when an attempt is made by BRU to write to an archive which is write- protected. The reason for the warning is indicated by the UNIX error message. This warning may also occur if the backup device does not respond properly when BRU attempts to open the device for writing. BRU is "faked out" and thinks that the device is write-protected. This often occurs with on the first attempt to write to a SCSI device. Try repeating the command. If BRU works successfully, this message can be safely ignored.

[W109] warning - assuming end of volume volnum (unknown size)

BRU encountered an unrecoverable read or write error before reaching the end of an archive while reading or writing a volume of unknown size. BRU may have actually reached the end of the volume, or BRU may have simply reached a bad spot on the media, which BRU cannot proceed past. Because BRU does not know the media size, BRU has no way of knowing the difference, hence, the warning message. If no other warnings, or errors, occur, this warning is benign.

[W110] warning - found volume volnumX, expecting volnum

BRU was expecting to find volume `volnum` and it encountered a different volume. Remove the volume and replace it with the correct volume.

[O111] *** OBSOLETE MESSAGE NUMBER ***

[W112] warning - volume not part of archive created archivedate

BRU received the correct volume number, but the date of the volume differs from the current archive. Generally, this warning occurs when the wrong tape is inserted while attempting to extract an archive.

[A113] alert - all data currently on devicename will be destroyed

When the brutab entry for a device includes the `qfwrite` boolean value, this message will be issued on the first write to the first volume placed in that device, and BRU will wait for confirmation to continue. In devices which might share both mounted and unmounted media, this prevents inadvertently overwriting media which may have been left in the device by mistake.

[I114] *** OBSOLETE MESSAGE NUMBER ***

[A115] *** OBSOLETE MESSAGE NUMBER ***

[I116] *** OBSOLETE MESSAGE NUMBER ***

[I117] don't know how to rewind archive device

BRU doesn't know how to rewind the present archive device.

[A118] rerun with "-b bufsizek" argument

Re-run your BRU command with the specified buffer size

[Q119] action filename: please confirm [y/n/g]

BRU is waiting for confirmation of the given action. The **-w** flag was specified on the command line.

[Q120] query options [default: Option] >>

General message used to prompt user with various messages.

[A121] load volume volnum - press ENTER to continue on device device name

[W122] *filename*: warning - too large under current ulimit, not extracted

The size of filename exceeds the current ulimit. Set ulimit to a larger value and try again.

[E123] ulimit call failed to set maximum file size limit to blkcount blocks

BRU was unable to set the ulimit to a larger value.

[W124] warning - no double buffering support included in this version

Your version of BRU does not support double buffering. BRU reverts to normally buffered I/O.

[W125] warning - shared memory does not appear to be working in your kernel

Verify the settings in you kernel. Contact your UNIX provider for help.

[E126] problem sending message to other process

While in double-buffer mode, BRU could not communicate with the child process.

[E127] problem receiving message from other process: errno = code, description

While in double-buffer mode, BRU could not communicate with the child process.

[W128] *filename*: warning - file contents changed while archiving

BRU found that the file it was backing up had been changed. This can happen during live system backups. BRU does not lock files when reading. If a file is modified while BRU is reading it, this message will occur.

[W129] *** OBSOLETE MESSSAGE NUMBER ***

[W130] warning - I/O error on first block

BRU cannot read or write the first block of the archive. Verify that the given device has a tape in the drive and that the tape has not been ejected.

[W131] warning - archive device may need "ignoreclose" flag set in brutab entry

We have found that some tape drives require this setting in the brutab file. For additional information on this settings please refer to chapter 4 of this User's Guide

[W132] warning - media size automatically adjusted to size

This message is issued when BRU starts writing to a new volume and has changed the media size to a value which is different than the size originally specified. This can occur if BRU encountered the end-of-tape sooner than expected. For example, the media size was specified as 150M, but BRU hit the end-of-tape (on the first tape) at 120 Mb. BRU will then ask for the next volume and adjust the media size (for the second tape) to a value slightly less than 120 Mb.

[W133] warning - no entry for device devicename in brutabfile

BRU could not locate an entry in your brutab file for the given device.

[E134] internal error errcode - failed self consistency and portability checks

This indicates that the BRU executable file has been damaged or the BRU version is not the correct one for your system. This usually occurs when BRU cannot successfully determine your timezone offset from GMT or receives a bad value for the current date and time from a tzset function call.

[E135] path beginning with filename too large (maxlen characters max)

The specified filename exceeds the system limit.

[W136] filename1: warning - link to filename2 broken, saved as duplicate

The character length of the link name **filename2**, was too large to store in the file header block.

[W137] warning - wait failed: errno = code, description

During double-buffering, an error occurred while waiting for a child process.

[E138] error - unable to format device devicename

BRU could not format the device. It may be write protected, or command specified by **fmtcmd=** failed when attempting to format the device.

[E139] error - timed out during execution of pid pidnum

BRU attempted to execute an external program and the process timed out.

[E140] error - unable to read include/exclude pattern file: filename

The specified include/exclude pattern contains an error. BRU could not read the file.

[W141] warning - bruxpat pattern: pattern, errmsg

The specified include/exclude pattern contains an error. The reason for the error is indicated by **errmsg**. Edit the bruxpat file to correct the problem.

[W142] -E option ignored with -c, -e, -i or -t options

This is an illegal command option for the given modes (**-c -e -i** or **-t**)

[I143] rewinding volume [volnum] to begin autoscan

BRU prints this informational message to let you know it is rewinding volume number **volnum** to begin the **AUTOSCAN**. Rewinding may sometimes take a substantial (several minutes) length of time, it depends on the speed of your tape drive.

[I144] begin autoscan of volume [volnum]

BRU prints this informational message to let you know it has begun the auto-scanning of volume number **volnum**. Auto-scanning may sometimes take a substantial (several minutes to over an hour) amount of time, depending on the speed of your archive device.

[E145] autoscan checksum error at block blocknum (kbsize)

BRU detected a problem while performing a checksum validation during the **AUTOSCAN** phase. It could be an indication of tape hardware problems, dirty tape heads, an improper BRU buffer size, tapes with the wrong density, tapes that need to be retensioned, or tapes that are simply worn out. If this message occurs at the beginning of a tape (**blknum** is less than 10), it may indicate that your tape drive (or device driver) contains a bug and returned control to BRU before it finished rewinding. BRU tried to start the **AUTOSCAN**, but was unable to read the tape (because it was still rewinding). Often this problem can be fixed by setting the **maxrewindtime** parameter for your device in the **/etc/brutab** file. A setting of **maxrewindtime=300** seems to work for most tape drives. Sometimes this message occurs with tape drives that are confused by BRU's overwrite-protect feature. If overwrite-protect is enabled, BRU attempts to read the tape before it tries to write. Some tape drives cannot handle this. If this is the case, edit

the **/etc/brutab** file and disable the global brutab parameter "**#+ OVERWRITE PROTECT**" by removing the "**+**" sign or by deleting the line.

[E146] unable to get memory (bytecount bytes) needed for autoscan buffer

> BRU is unable to allocate enough memory to create an **AUTOSCAN** buffer. This message usually indicates that the buffer size (the **bufsize** or **asbufsize** parameters in **/etc/brutab**) is too large. It may also be caused by a shortage of memory or by system problems.

[W147] cannot do autoscan - device 'devicename' has "flagname" flag set in brutab

> This usually indicates that the **norewind** or **noautoscan** flags are set in the device entry in your
> **/etc/brutab** file.

[E148] autoscan read error at block blknum (kbsize): errno = code, description

> BRU received an error when attempting to read an archive during the **AUTOSCAN** phase. The reason for the error is indicated by the UNIX error message. This message may be an indication of tape hardware problems, dirty tape heads, an improper BRU buffer size, tapes with the wrong density, tapes that need retensioning, or tapes that are simply worn out. Often, this message occurs along with [E145]. Refer to its description for more information.

[W149] autoscan detected errors - media or hardware may be bad

> This message indicates that errors or warnings were detected during the **AUTOSCAN** phase.

[I150] autoscan of blkcount blocks on volume [volnum], time, speed Kb/sec

[L151] autoscan of blkcount blocks on volume [volnum], time, speed Kb/sec

> These messages are completion messages for the **AUTOSCAN** pass.

[E152] error - timed out trying to open filename

> BRU timed out while trying open a device or file.

[E153] error - timed out trying to read

BRU timed out while trying read form a device or file.

[E154] error - timed out trying to write

> BRU timed out while trying to write to a device.

[E155] error - memory fault (SIGSEGV)

> BRU received a **SIGSEGV** signal from your OS. This is generally indicative of a memory issue in your system.

[E156] error - memory fault (SIGSEGV) in child process

> BRU received a **SIGSEGV** signal from UNIX on one of its child processes. This is generally indicative of a memory issue in your system.

[E157] error - received terminate signal (SIGTERM)

> BRU received a **SIGTERM** signal from your OS.

[E158] error - received quit signal (SIGQUIT)

> BRU received a **SIGQUIT** from your OS.

[E159] error - received interrupt signal (SIGINT)

> BRU received a **SIGINT** signal from your OS.

[E160] error - received hangup signal (SIGHUP)

> BRU received a **SIGHUP** signal from your OS.

[E161] error - received strange signal (signame)

> BRU received an unexpected signal.

> For more information on the various signals please refer to your operating systems documentation or contact the OS manufacturer.

[W162] warning - unable to open execution log file 'logfilename': errno = code, description

> BRU was not able to open the **bruexeclog** file. This could be due to a permissions problem or the location of the file is not what BRU expected. With every BRU command or operation BRU will attempt to make an entry to the **BRUEXECLOG** file.

[L163] START (info), CMD = 'cmdline'

The entry in you **BRUEXECLOG** and to your screen shows that actual command that was started during your BRU process. It also shows you the release and version of BRU you are running.

[L164] START - child process for double-buffering

[L165] FINISH - warncount warnings, errorcount errors, exit code = exitcode

[L166] starting volume volnum on device "devicename"

[L167] device = devicename, buffer = bufsizeK bytes, media size = size

[E168] *** OBSOLETE MESSSAGE NUMBER ***

[E169] error - bad argument for -T option

The user entered BRU options in an illegal combination.

[E170] error - illegal combination of mode options (-cdeghitx)

The user entered BRU options in an illegal combination.

[W171] warning - needs to be owned by root and have suid bit set

[W172] warning - cannot open device 'devicename' to do autoscan

BRU could not open **devicename** to perform **AUTOSCAN** verification.

[E173] error - exceeded warning count limit of maxwarn

BRU exceeded the maximum number of allowed warnings and terminated. The maximum number of warnings can be changed with the **BRUMAXWARNINGS** environment variable.

[E174] error - exceeded error count limit of maxerror

BRU exceeded the maximum number of allowed errors and terminated. The maximum number of errors can be changed with the **BRUMAXERRORS** environment variable or the **BRUMAXERRORS GLOBAL BRUTAB** setting - see Chapter 4.

[E175] error - bad pattern match on: {/etc/bruxpat}, entry

This would and should be superseded by **[W141].**

[E176] error - bad raw block size = blksize, cannot extract raw file 'filename'

BRU cannot restore or extract the given raw file from the archive. The information specified in the **BRURAW** file has an incorrect block size defined.

[W177] warning - specified size is too large, try setting size to newsize Kbytes

BRU has determined that the size given for the device you are using as an archive deice is to high. BRU has suggested that you use a smaller size as the one specified in this warning message.

[L178] rewinding volume [volnum] to begin autoscan

BRU writes this message to the **bruexeclog** to time stamp when it began to rewind volume **volnum** prior to starting an **AUTOSCAN**. Rewinding may take a substantial (several minutes) amount of time.

[L179] issued reset cmd 'cmdstring'

BRU has reset the device as specified by the reset command you defined in the **BRUTAB** file

[W180] warning - reset cmd error: errno = code, description

BRU received a UNIX error message when attempting to reset the device with the given reset command as specified in the **BRUTAB** file

[I181] read/wrote blkcount blocks on volume [volnum], time, speed Kb/sec

This reports the number of blocks written/read during the current BRU process. This will also indicate the speed at which the process ran by showing you the time and speed in Kb/sec.

[L182] read/wrote blkcount blocks on volume [volnum], time, speed Kb/sec

This reports the number of blocks written/read during the current BRU process. This will also indicate the speed at which the process ran by showing you the time and speed in Kb/sec

[W183] skipped autoscan of volume [volnum]: reason

BRU prints this message to let you know it has skipped the **AUTOSCAN** of volume **volnum**. The reason is indicated as part of the message.

[I184] waiting time seconds to finish rewind

Reports that BRU is waiting a certain amount of seconds before attempting to complete the process

[L185] `waiting time seconds to do rewind`

Reports that BRU is waiting a certain amount of seconds before attempting to complete the process

[L186] `using 'rshname' as remote shell`

BRU will use the given file name as it's remote shell

[E187] `unable to execute remote shell 'rshname'`

BRU is unable to use the given filename as the remote shell. Verify that it exists and that you have the proper permission set.

[E188] `cannot find remote shell to execute`

BRU can not execute the remote shell. Verify it is in the correct location and that you have proper permissions set.

[A189] `filename is not a device`

You have attempted to write to a file name. BRU is reporting that this is not a device but a file.

[A190] `file filename already exists`

BRU is reporting that you are attempting to write to a file that already exists on the system.

[E191] `error - compression buffer too large, cannot allocate kbcountk bytes`

The given **`ZBUFSIZE`** in your **`BRUTAB`** file is too large for your system to support. You should change the **`ZBUFSIZE`** setting in your **`BRUTAB`** file to a smaller number then re-attempt your process.

[E192] *`filename`*`: compression error (errmsg), data is corrupted`

An error occurred while BRU was compressing filename, the reason for the error is indicated by **`errmsg`**. The file was not backed up properly and the archived file contains errors. Normally, this error is caused when BRU attempts to back up a file that was changing (like a database file). For information, refer to "Live System Backups" in Chapter 9. In a few rare occurrences, this error has been caused by disk controller hardware failures or by corrupted filesystems.

[E193] *`filename`*`: decompression error (errmsg)`

An error occurred during the restore and decompression of a compressed file. The **`errmsg`** will indicate what the problem is with the file.

[E194] *`filename`*`: warning - file was not backed up: errno = code, {description}`

This message is issued when BRU encounters a problem on its first attempt to read filename. Often, this is caused by improper permissions. If filename is part of an NFS-mounted filesystem or the **`/etc/export`** file (and BRU is running as root), this message can occur if BRU is unable to access the file due to insufficient root permissions. You may need to modify your network security parameters by editing your **`/etc/hosts.equiv`** or **`.rhosts`** file. Configuring network security can get complicated (and all networks are different), so refer to your UNIX network documentation for details.

[Q195] `enter new device name [default: devicename] >>`

BRU is requesting that you enter a new device name or to continue the operation on the default device.

[E196] `error - attempt limit exceeded ... BRU terminated`

BRU has reached it's maximum numbers of errors given in the **`BRUTAB`** file.

[E197] `error - illegal device name ... BRU terminated`

BRU has attempted to write to a device that is not stated in the BRUTAB file. The setting in BRUTAB of **`BRUTABONLY=YES`** has forced this error. By changing the default value to **`NO`** for this setting you will be able to write to this device.

[W198] `try using a smaller buffer size (like bufsizeK)`

BRU is attempting to write to a device that is responding with errors. BRU has determined this write error as having to large of a buffer size setting set for the given device. By using the **`-b`** option in your command line and change the buffer size to a smaller size BRU might be able to complete the operation.

[A199] `OVERWRITE PROTECT: volume is from archive written on date`

BRU has found that the date of this archive is within it's overwrite protect setting.

[A200] insert another volume and press ENTER to continue

Query message asking for a different tape.

[E201] user entered Q => QUIT in routine routinename

A user entered Q to quit the operation on one of BRU query to the terminal.

[I202] switching to next device

This is an information message stating that BRU is switching to a different tape drive. This message is sent during a device cycling operation

[I203] device cycling discontinued

BRU has stopped device cycling.

[W204] *filename filename1* **too long, changed to** *filename2*

This message occurs when BRU restores a file with a name that is longer than 14 characters (the maximum on older UNIX systems). BRU automatically renames the file and shortens it to 14 characters. If your system supports filenames longer than this, this behavior can be suppressed. Simply add the global brutab parameter "**#+ MAXFILENAMELEN=255**" to the beginning of the **/etc/brutab** file.

[I205] *filename* **is an existing directory**

The given file name in **filename** is a directory

[W206] regular expression error, string

[E207] error - failed MOUNTCMD/UNMOUNTCMD of volume volnum on device device name (exit code = #)

BRU attempted to run the specified **MOUNT** and **UNMOUNT** commands as was not able to complete the operation. The device and volume number is listed in the error message.

[W208] warning - autoscan buffer size adjusted to newbufsizeKb

BRU found that the given buffer size for the **AUTOSCAN** process had to be changed to complete the process. If you see this warning message you should adjust your **BRUTAB** buffer size setting to match the given size for the warning message.

[W209] warning - skipped archive file filename

BRU has skipped the file listed on the warning message.

[W210] warning - could not rewind device

BRU could not rewind the given device.

[E211] unable to write nullcount end nulls

When closing the archive for a specific write, BRU was unable to write the entire closing block of data. This can be caused by low memory, no disk space (of writing to a disk archive), or a write failure on a tape drive.

[E212] BRU terminated, media may be write protected or wrong density

BRU could not continue to write to the device. The reason for this error could be that the configuration of the tape drive size was wrong or that you are using a different size tape on this device.

[W213] could not read password file

BRU could read your password file. This can be caused by a permissions problem. Try running the process again as root and see if that resolves the warning.

[W214] could not read group file

BRU could not read your group file. This can be caused by a permissions problem. Try running the process again as root and see if that resolves the warning.

[I215] translating filename to filename

This is a status message informing you that the filename is being translated to the new filename

[L216] translating filename to filename

This is a log entry you will not see this message echoed to the screen

[W217] warning - filename translates to "null"

BRU is reporting the given filename in your translate file is translating to null

[E218] error - unable to read translate table file: filename

BRU is reporting that it can read your translation file. Make sure that the file is in an ASCII format and is in the given directory stated in your **BRUTAB** file

[E219] translation table error 'ERROR TEXT'

If you have specified a translation table via the -T command line option, BRU is unable to read the contents of the specified file. The '**ERROR TEXT**' will describe the exact error encountered.

[O220] * OBSOLETE MESSAGE NUMBER ***

[E221]** *filename* **: cannot overwrite directory with file**

BRU can not overwrite the directory with a file.

[E222] *filename* **: cannot overwrite file with directory**

BRU can not overwrite the file with a directory.

[E223] unable to inspect "norewind" device

The given device has the "**norewind**" option set in **BRUTAB**. With this option set BRU can not rewind the device and perform the **AUTOSCAN** option.

[E224] error - invalid -u argument "%s"

Invalid argument with the **-u** option. Valid options are (**a b c d i p r f**)

[E225] error - unable to execute MOUNTCMD="filename"

BRU can not run the given **MOUNTCMD** or it is attempting to run a command that is illegal.

[W226] "device" : device open when UNMOUNTCMD was called (loc=%d)

The device was open with a different request when BRU called the **UNMOUNTCMD**

[W227] "filename": warning - error setting owner/group on symbolic link: errno = number, {description}

BRU could not set the owner/group information for the specified file. The UNIX error number and description, descried the problem.

[E228] error - unable to read raw-partition file: "filename"

BRU could not read the given **BRURAW** file. Make sure that the file has the correct permissions and is in the given directory.

[E229] error - no raw-device description in bruraw table: "filename"

There was no information provided in the **BRURAW** file.

[W230] warning - invalid raw-device specified (& ignored): "filename"

A invalid device was specified in the **BRURAW** file BRU has now ignored that device and is not backing up the given device.

[W231] warning - problem reading label file: "filename"

BRU could not read the given file to create a label from. Make sure that the file exist and that

[W232] *filename***: warning - file may be locked**

BRU is reporting that the given file name might be a locked file. BRU can not read the file and this might cause BRU to terminate if you have set a low setting for the maximum number of warnings.

[W233] warning - unable to read smart-restore pattern file: filename

BRU is not able to read your given Smart-Restore file. Verify that the file is in the correct ASCII format and has the correct permissions set.

[W234] warning - smart-restore parse error '%s' on line '%s'

An error occurred while parsing your **/etc/smartrestore** file.

[W235] warning - problem creating archive catalog: filename

BRU could create its archive catalog file in the given directory specified in the **GLOBAL BRUTAB BRUTEMP=directory**

[W236] * OBSOLETE MEGSSAGE NUMBER ***

[W237] *** OBSOLETE MEGSSAGE NUMBER ***

[E238] error - invalid -Q argument: "%s"**

The argument that you passed with the **-Q** option is not valid. Arguments for **-Q** must immediately follow the **-Q** (i.e.: **-QL**) and be separated from the other options on the command line by at least one white space character.

[E239] error - invalid -U argument: "%s"

The only arguments for **-U** are numeric (0, 1, 2, etc).

[E240] error - unable to read, tape may be incompatible

[W241] warning - cannot translate "%s" to "%s"

Attempts to translate filenames during a restore were not allowed. This is probably a permissions problem.

[A242] enter label for volume %d:

[E243] error - previous UNMOUNTCMD on device "%s" failed

[W244] warning - left %d temporary files, check removelog file "%s"

During a restore, BRU ran into a number of open/in use files. These files were copied to a temporary directory and the tape version was restored. You may remove the temporary files by running the shell script displayed.

[I245] "%s": skipped file, %s

The file listed was skipped in this operation because of the reason given.

Appendix N
OS X LTFS Support Requirements

The following are the requirements for enabling LTFS (Linear Tape File System) support on your Mac OS X system:

- LTFS-compatible LTO-5 or later tape drive
- Your Tape Drive Hardware manufacturer's LTFS driver suite

The following items are *not support* by LTFS:

- PowerPC systems
- Systems running earlier versions of OS X prior to 10.6.5
- Systems that do not support PCIe® expansion (Fibre Channel or SAS) - because you can't connect an LTO drive.
- LTO drives connected via iSCSI

TOLIS Group is not able to provide drivers for all vendors' LTO drives. Please contact your LTO vendor to request the LTFS drivers required for your drive.

LTFS Command Line Tools

There are three command line tools provided for use with LTFS volumes. The functionality of these tools is duplicated in the ArGest Backup LTFS Manager interface. The commands are:

mkltfs   used to format LTO-5 tapes for use as LTFS volumes

ltfs   used to mount the LTFS formatted tapes

unltfs   used to remove LTFS formatting from tapes

ltfsck   used to validate mounted LTFS volumes and perform generation rollback

You do not need admin access to use these tools under OS X.

For completeness, the command line tool options for the LTFS 2 release are outlined below.

```
mkltfs
usage: mkltfs <options>


Available options are:
  -d, --device=<name>      Tape device (required)
  -s, --tape-serial=<id>   Tape serial number (6
             alphanumeric ASCII characters)
  -n, --volume-name=<name> Tape volume name (empty by
                           default)
  -r, --rules=<rules>      Rules for choosing files to
                           write to the index partition.
                           The syntax of the rule argument
                           is:

       size=1M
       size=1M/name=pattern
       size=1M/name=pattern1:pattern2:pattern3
                           A file is written to the index
                           partition if it is no larger
                           than the given size AND matches
                           at least one of the name
                           patterns (if specified). The
                           size argument accepts K, M and G
                           suffixes. Name patterns may
                           contain the special characters
                           '?' (match any single character)
                           and '*' (match zero or more
                           characters).
  -o, --no-override        Disallow mount-time data
                           placement policy changes
  -f, --force              Force reformat of existing LTFS
                           volume (normally prevented)
  -q, --quiet              Suppress progress information
                           and general messages
  -t, --trace              Enable function call tracing
  -h, --help               This help
  -p, --advanced-help      Full help, including advanced
                           options
  -i, --config=<file>      Use the specified configuration
                    file (default:
/Library/Frameworks/LTFS.framework/Versions/Current/etc/
ltfs.conf)
  -e, --backend=<name>     Use the specified tape device
                           backend (default: ltotape)
  -b, --blocksize=<num>    Set the LTFS record size
                           (default: 524288)
  -c, --no-compression     Disable compression on the
                           volume
  -x, --fulltrace          Enable full function call
                           tracing (slow)
```

```
Usage example:
  mkltfs --device=0 --rules="size=100K"
  mkltfs --device=0 --rules="size=1M/name=*.jpg"
  mkltfs --device=0 --rules="size=1M/name=*.jpg:*.png"
```

ltfs

```
usage: ltfs mountpoint [options]

general options:
    -o opt,[opt...]         mount options
    -h   --help             print help
    -V   --version          print version

FUSE options:
    -d   -o debug           enable debug output (implies -f)
    -f                      foreground operation
    -s                      disable multi-threaded operation

LTFS options:
    -o config_file=<file>  Configuration file (default:
/Library/Frameworks/LTFS.framework/Versions/Current/etc/
ltfs.conf)
    -o work_directory=<dir> LTFS work directory
                            (default: /tmp/ltfs)
    -o atime                update XML if only access times
                            have changed
                            (default: do not update)
    -o tape_backend=<name>  tape backend to use
                            (default: ltotape)
    -o iosched_backend=<name> I/O scheduler implementation
                            to use (default: unified, use
                            "none" to disable)
    -o umask=<mode>         Override default permission mask
                            (3 octal digits, default: 000)
    -o fmask=<mode>         Override file permission mask
                            (3 octal digits, default: 000)
    -o dmask=<mode>         Override directory permission
                            mask (3 octal digits,
                            default: 000)
    -o min_pool_size=<num>  Minimum write cache pool size.
                            Cache objects are 1MB each
                            (default: 25)
    -o max_pool_size=<num>  Maximum write cache pool size.
                            Cache objects are 1MB each
                            (default: 50)
    -o rules=<rules>        Rules for choosing files to
                  write to the index partition.
                            The syntax of the rule argument
                            is:
    size=1M
    size=1M/name=pattern
    size=1M/name=pattern1:pattern2:pattern3
                            A file is written to the index
                            partition if it is no larger
                            than the given size AND matches
```

```
                         at least one of the name
                         patterns (if specified). The
                         size argument accepts K, M and G
                         suffixes. Name patterns may
                         contain the special characters
                         '?' (match any single character)
                         and '*' (match zero or more
                         characters).
    -o quiet             Disable informational messages
    -o trace             Enable diagnostic output
    -o fulltrace         Enable full call tracing

    -o eject             Eject the cartridge after
                         unmount
    -o sync_type=<type>  Specify sync type (default:
                         time@5)
       <type> should be specified as follows
       time@min: LTFS attempts to write an index each
               min minutes
               min should be the decimal number from 1 to
               153722867280912930 (default: min=5
       close:    LTFS attempts to write an index when a file
               is closed
       unmount:  LTFS attempts to write an index when the
               medium is ummounted
    -o force_mount_no_eod  Skip EOD existence check when
                         mounting (read-only mount)
                         Only use for a CM corrupted
                 medium
    -a                   Advanced help, including
                         standard FUSE options


  LTOTAPE backend options:
    -o devname=<dev>       tape device (default=0)
    -o log_directory=<dir>  log snapshot directory
             (default=/Users/tjones/Library/Logs/LTFS)


  IOKIT backend options:
    -o devname=<dev>       tape device (default=0)


  FILE backend options:
    -o devname=<dev>       LTFS emulation directory
(default=/tmp/ltfs/tape)
    -o file_readonly       Emulate operation in read-only
                         mode
```

unltfs
```
Usage: unltfs <options>

where:
        -d, --device=<name> specifies the tape drive to use
        -y, --justdoit       omits normal verification
                  steps, reformats without further
                  prompting
        -e  --eject          eject tape after operation
                  completes successfully
        -q, --quiet          suppresses all progress output
        -t, --trace          displays detailed progress
        -h, --help           shows this help
        -i, --config=<file> overrides the default config
                  file
        -b, --backend        specifies a different tape
                  backend subsystem
        -x, --fulltrace      displays debug information
                  (verbose)
```

ltfsck

Usage: ltfsck [options] filesys

filesys   Device file for the tape drive

Available options are:
```
 -g, --generation=<gen>    Specify the generation to roll
                           back
 -t, --time=<time>         Specify the time to roll back
                           Time value should be
                           "yyyy-mm-dd HH:MM:SS.nanosec"
 -r, --rollback            Rollback to the point specified
                           by -g or -t
 -n, --no-rollback         Do not roll back. Just verify
                           the point specified by -g or -t
                           (Default)
 -f, --full-recovery       Recover extra data blocks into
                           directory _ltfs_lostandfound
 -z, --deep-recovery       Recover cartridge which has EOD
                           missing NB Some blocks may be
                           erased but recover to final
                           unmount point which has an index
                           version "2.0.0" or earlier
 -l, --list-rollback-points List rollback points
 -j, --erase-history       Erase history at rollback
 -k, --keep-history        Keep history at rollback
                           (Rollback)
 -q, --quiet               Suppress informational messages
 -a, --trace               Enable diagnostic output
 -h, --help                This help
 -p, --advanced-help       Full help, including advanced
                           options
 -i, --config=<file>       Use the specified configuration
                           file
(default: /Library/Frameworks/LTFS.framework/Versions/
Current/etc/ltfs.conf)
 -e, --backend=<name>      Override the default tape
                           device backend
 -x, --fulltrace           Enable full function call
                           tracing (slow)
```

Appendix O
LTFS Support

ArGest Backup supports formatting, mounting, and catalog maintenance of LTFS (Linear Tape File System) volumes on OS X 10.11 and newer systems. However, you must manually enable the LTFS support and agree to that you understand that TOLIS Group does not provide ANY support for the LTFS installation on your system. This functionality is provided as a convenience feature and is not a mainstream feature or selling point for ArGest Backup.

LTFS is a tape standard being promoted by the LTO organization members to allow the use of LTO-5 and later tapes in a "disk-like" fashion. The goal is to provide a self-defining format that is compatible across multiple platforms without the requirement for 3rd party software. While this is a worthy goal, the reality of using tape in a disk-like fashion will never make this as transparent as the vendors would like.

One of the weaknesses of the LTFS operation under OS X is that there is no way to locate files on LTFS volumes that are not currently mounted on the system. However, with ArGest Backup's LTFS import capability, you can import LTFS tapes - regardless of the creating source - and search them in the same manner as your normal ArGest Backup tapes using the Restore panel.
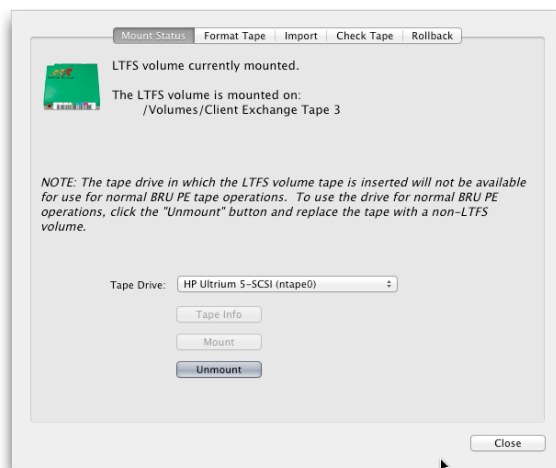

LTFS Manager Dialog

The ArGest Backup LTFS Manager Dialog allows you to format LTO tapes for use as LTFS volumes, mount of unmount LTFS volumes on the OS X system, and import the contents of an LTFS volume so that it can be accessed even if the LTFS tapes are offline (not mounted on the system).

To Open the LTFS Manager, use the "View" -> "LTFS Manager" menu or the ⌥⌘-F (Opt-Cmd-F) keystroke combination.

If you are starting fresh, refer to Appendix N for more information on the prerequisites for LTFS support under OS X.

LTFS Preferences
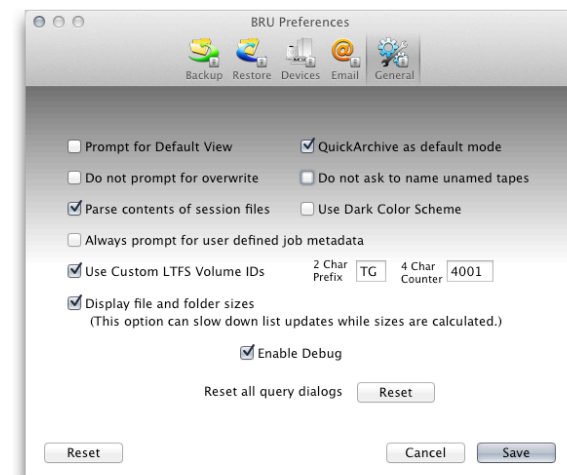
LTFS Volume Defaults

By default, ArGest Backup will format tapes using a volume ID (6 character identifier) using the format of TG0001 and incrementing the count to 9999 as tapes are formatted. If you wish to provide custom volume IDs for your LTFS tapes, open Preferences (Cmd-,), select the **General** tab and check the "***Use Custom LTFS IDs***" checkbox. The 2 character Prefix value may be any alpha-numeric value (0-9, aA-zZ). The 4 character Counter value *MUST* be 4 numeric characters (0000-9999). The two will be combined to create the volume ID for the tape when it is formatted. For example, a prefix of "TG" and a counter of "9001" would become an ID of "TG90001", "TG9002", "TG9003", and so on to allow for 999 uniquely identifiable tapes.

Formatting an LTFS Tape

Open the LTFS Manager and select the "Format" tab.
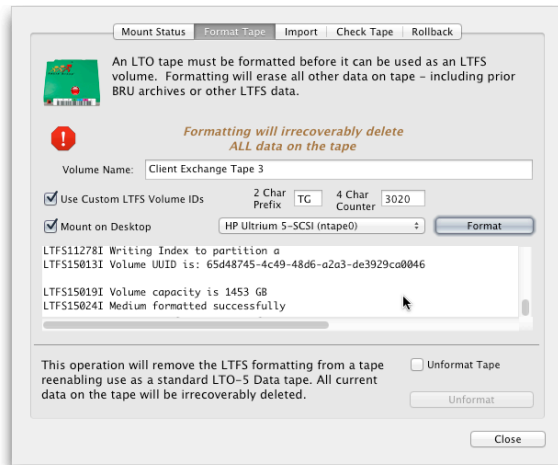
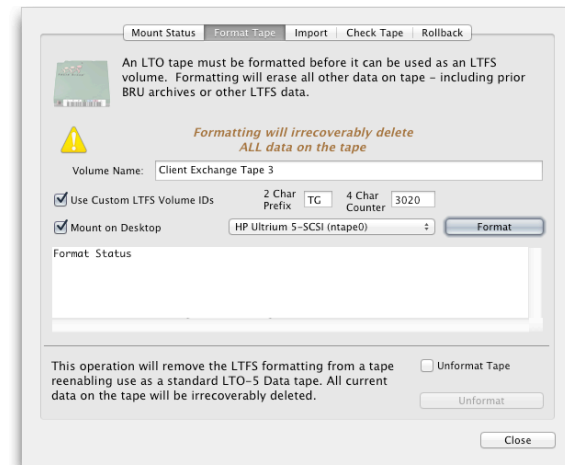**LTFS Manager - Format Tab**

Important: Tapes formatted to the dual partition LTFS format are NOT usable as standard ArGest Backup backup or archival media.

If you have more than one LTO-5 or LTO-6 drive connected, select the drive you'll be using for this format operation.

If you wish to mount the tape after it is formatted, check the "***Mount on Desktop***" checkbox. With this option set, the tape will be mounted for use once the format is completed. If you do not set this option, the tape will be formatted and prepared and then ejected allowing you to format additional tapes.

The name that you provide for the volume in the "Volume Name" field will also be the name under which the tape will be mounted on your system. This name can be any text up to 128 characters. However, if you wish to maintain the widest level of platform compatibility, you should stay away from special characters such as *, $, &, %, |, [,], {, }, (,), !, @, #, ~, `, \, and /.

Once you've set your options, insert a blank tape (any data on a previously used LTFS or BRU tape will be permanently erased) and click the "***Format***" button

**LTFS Manager - Format Options Set**

Formatting an LTFS tape takes approximately 2 minutes until the tape is mounted and ready to use or ejected.

If no tape is available, you will be prompted to insert a tape. Once the formatting begins, it cannot be aborted since as soon as the drive starts the partitioning operation, the operation must complete of the tape will be left in an unusable state - for both LTFS and ArGest Backup.
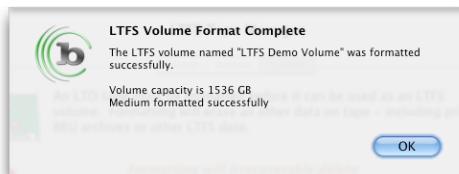

LTFS Manager - Formatting Status

After the format operation is completed, the tape will be mounted and the icon and finder starting details will be applied to the volume.
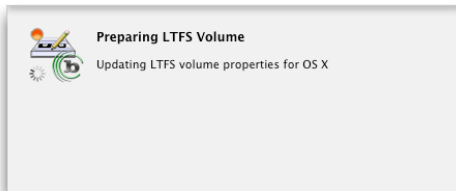
LTFS Manager - Preparing Volume

Once the preparation is complete, if you have not selected to mount the volume on your desktop, the tape will be ejected. If you selected to mount the volume, it will be visible on your desktop with the name you assigned in the options dialog and ready to use.
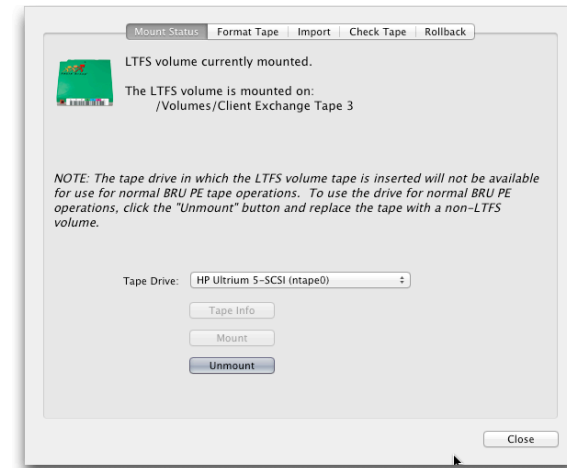

LTFS Manager - Format Complete

In either case, the completion dialog will display with the information about the volume indicated.

Mounting and Unmounting LTFS Volumes

LTFS Manager - Volume Mounted

The LTFS Manager dialog's "***Status***" tab displays the current mount state for the selected drive. If an LTFS volume is mounted, the mount point is displayed as shown above.

Mounting and unmounting a volume is a simple button click action. No further input is required as the LTFS tape has all of the information required for mounting included in its index.
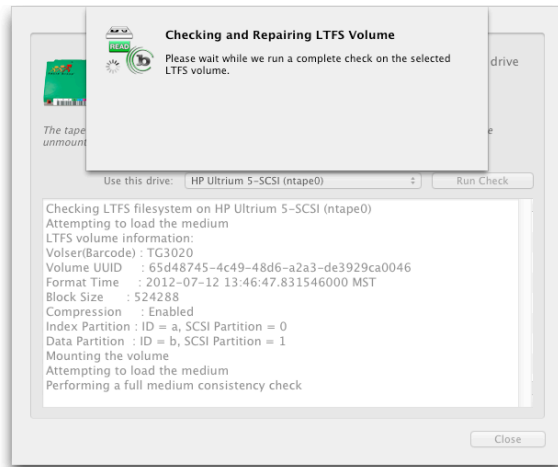
Clicking the "Tape Info" button will display the pertinent info about the current unmounted tape in the selected drive.

> ***Note***: *If your tape drive has an LTFS volume mounted, that tape drive cannot be used for normal ArGest Backup operations. If you need to use the drive for ArGest Backup tape operations, you must first unmount the LTFS volume and insert a non-LTFS tape.*

Checking and Repairing LTFS Volumes

There are situations that can cause an LTFS volume to become unaccessible on your system (the mount will fail). These include loss of power on the tape drive with a tape mounted, a system crash with a tape mounted, and other unexpected situations.

ArGest Backup provides a mechanism for checking and repairing LTFS volumes via the "Check Tape" tab.

Checking and Repairing LTFS Volume

Please wait while we run a complete check on the selected LTFS volume.

The tape
unmount

Use this drive: HP Ultrium 5–SCSI (ntape0)   Run Check

```
Checking LTFS filesystem on HP Ultrium 5–SCSI (ntape0)
Attempting to load the medium
LTFS volume information:
Volser(Barcode) : TG3020
Volume UUID    : 65d48745–4c49–48d6–a2a3–de3929ca0046
Format Time    : 2012–07–12 13:46:47.831546000 MST
Block Size    : 524288
Compression    : Enabled
Index Partition : ID = a, SCSI Partition = 0
Data Partition  : ID = b, SCSI Partition = 1
Mounting the volume
Attempting to load the medium
Performing a full medium consistency check
```

Close

LTFS Manager - Check Tape Tab

This function will verify both the data and index partitions and resolve any issues and differences between the two to make your tape accessible. The main condition is that if the error occurred after a Rollback has been performed, the Rollback may be undone in a manner that cannot be recovered with only the latest data available on the mounted media.

LTFS Volume Rollback
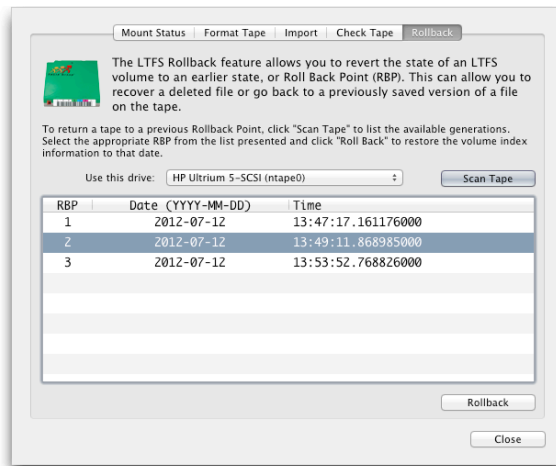
LTFS Manager - Rollback Tab

As you perform updates to an LTFS volume by either adding new data to an existing volume, overwriting existing files on the volume, or deleting files on the volume, the volume state prior to the change is saved and can be used as a "rollback point" on the LTFS volume. Because deletions and overwrites on an LTFS volume do not actually remove the original data from the tape, it is possible to reset the volume index pointers to the tape's status at the point in tie prior to the change.

When you scan a tape fro rollback points, the available index dates will be displayed and rolling back is as easy as selecting one of these points from the list and clicking the "Rollback" button.

Performing a rollback does not change any data on the tape, so it is possible to rollback to a previous state, retrieve a file, and then roll forward to the current state without losing the later changes to the volume.

Performing a rollback operation can take from 5 to 60 minutes depending on the number of index entries on the volume.

Importing LTFS Volume Information

Importing your LTFS volumes into ArGest Backup will allow you to access your LTFS volume information even if the tapes are offline (i.e.: not mounted or in storage).
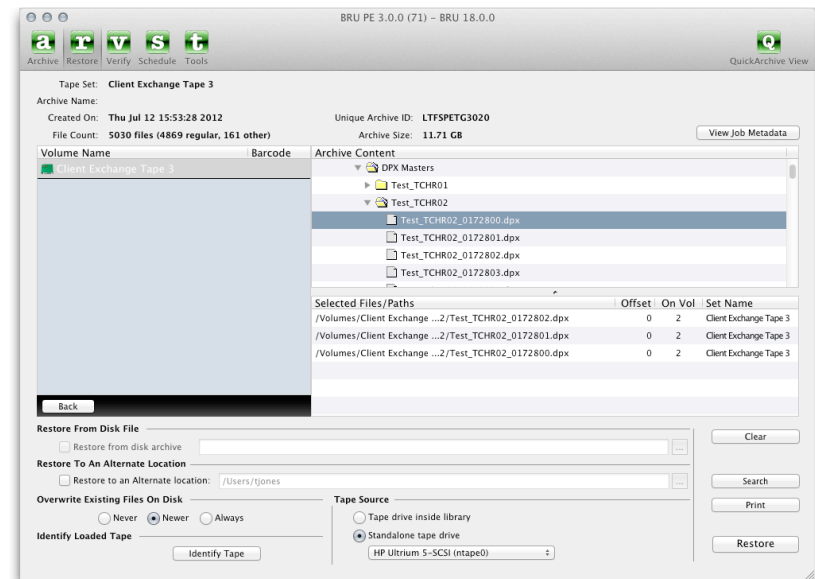
LTFS Volume Info In the Restore Panel

Once an LTFS Volume is imported, it will show up in the ArGest Backup Restore panel, can be searched and browsed just like a normal ArGest Backup archive as shown above. An LTFS volume that is known to ArGest Backup and currently online will have a normal LTFS icon displayed. Tapes that are not currently mounted will display a ghosted icon in the Restore list.

LTFS volume catalogs can be expanded and searched just like a normal ArGest Backup tape catalog.
To Import your LTFS volumes, use the "Import" tab on the LTFS Manager.
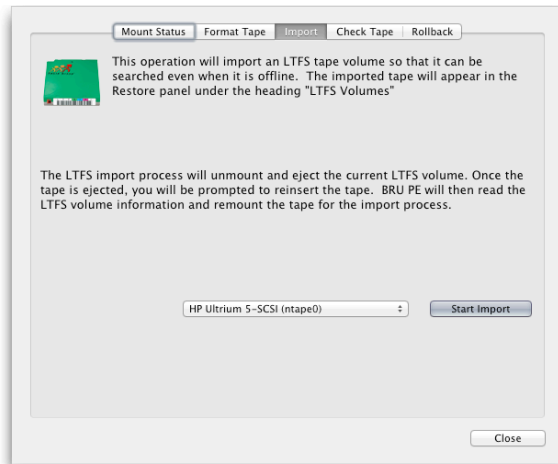
LTFS Manager - Import

Select the active LTO-5 tape drive and click the "Start Import" button. If no tape is currently inserted, you will be prompted to insert an LTFS volume.

If an LTFS volume is currently mounted, it will be unmounted and ejected to insure that the index partition is updated. You will be prompted to reinsert the tape into the drive.
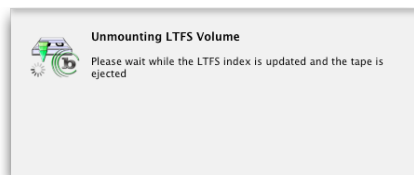
Scanning the Tape

Once a tape is recognized, it will be scanned to make sure that it is an LTFS volume. If it is an LTFS volume, the volume ID and the volume label is determined and the volume is mounted. BRU then creates a catalog of the data and adds it to the Tapes and Archives structure information.
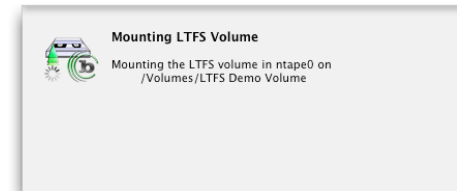
Mounting the LTFS Volume

Creating the Catalog Information

LTFS Volume Import Complete

Once the import is completed, the LTFS Volume and catalog information will be displayed in the Restore panel as shown above.

Note: If you add files to a previously imported LTFS volume, you must import it again to update the catalog information to include the newly added information.

Important LTFS Use Information

While the general marketing descriptions around the web about LTFS imply that you can use tape just like a disk, the reality is that there are many considerations when using a tape volume in this manner.

1. **LTFS is an Open Source project and comes with no direct support or warranty of usefulness**. From the readme information supplied with all LTFS implementations:

   `LTFS is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.`

   Be sure that you can live with tape access failures or platform incompatibilities before using LTFS for critical data storage.

2. **Do not use LTFS as a Live Filesystem.** LTFS is best for Nearline or offline archival storage and as such, LTFS volumes should be primarily used for archival or data sharing. While you can actively access files from an LTFS volume, we do not recommend that you do so. Simultaneously reading a file with an application and then saving the changes back to the tape results in delayed I/O and the loss of the original file's size in capacity on the tape since replacing an existing file does not free the original file's space on the tape, simple saving of changes will result in very slow application operation and a reduction in the apparent capacity of the LTFS volume (see item 8, below).

3. **Use Finder (and Finder replacements) sparingly.** Because the OS X Finder tries to retrieve file information for folders and files as they are displayed, every time you use Finder to access and expand folders on a mounted LTFS volume, you'll be required to wait while the tape is accessed to retrieve the freshly exposed files and folders. Also, while the Finder is seeking on the tape, all other Finder operations will be held (you will see the spinning beach ball or busy wait cursor) until the LTFS operation completes. This can impact other applications as well.

4. **Multiple Simultaneous Access is not recommended**. While OS X and the MacFUSE driver will not report an error if you try to simultaneously access multiple files from the tape, or perform read and write operations at the same time, the resulting tape partition changes and tape seeks will result in an extended wait for all operations to complete.

5. **Do not access the tape between writes If you are writing multiple segments of data.** If you are writing data from multiple sources in multiple write operations (i.e.: Copying the contents of multiple USB drives onto a single LTFS tape), it is faster to write each new segment of data to the LTFS volume with no further access of the tape between writes. This allows the drive to keep the tape at the current end of data (EOD) position so that the next segment can start writing immediately. If you access the LTFS volume between the write operations (i.e.: Open the volume in Finder), the drive will need to change to the Index partition and then seek back to the data being viewed. When you then start the next write operation, you will need to wait while the tape is positioned back to EOD before the write can continue.

6. **Additional write access will run more slowly than the first write**. In addition to item 4 above, performing multiple write sequences - especially after a tape has been unmounted and remounted - will slow down progressively as more data is added. This is caused by the actions required to update the index information and reposition the tape between the partitions and then move to EOD.

7. **Updating files does not release previously used space.** Since "overwriting" existing files on an LTFS volume does not free up the space used by the previous copy of the file, you will lose space on the volume equal to the size of the file that was replaced. After many updates, this can result in a tape that has far less than the rated 1.5TB of space. The only way to recover the space is to copy the current contents to another volume, reformat the LTFS volume and then copy the files to the freshly formatted volume.

8. **Do not use an LTFS volume to play back Music or Video files.** This is similar to the heading "Do not use an LTFS volume as a live filesystem." Because the MacFUSE layer only caches a select amount of data, using an LTFS volume as a music storage solution will result in choppy and failed playback as the tape seeks the non-cached portion of the currently playing song or video.

9. **LTFS doesn't support spanning volumes**. Unlike ArGest Backup, if you are writing to an LTFS volume, you must be aware of how much data you've written to the tape since LTFS, like disk, does not have a mechanism for prompting for a new tape. If you attempt to write more data than the tape will hold, you will simply get a "no space left on device" error and the write will fail. In the

same situation, ArGest Backup will safely write what it can onto the current tape and then prompt for an additional tape and complete the operation on the new volume.

10. **Do not depend on OS X system disk tools for information about an LTFS volume.** Because system tools like Finder's "Get Info...", "du", and "df" do not have logic for dealing with the compression on an LTO drive or the space lost to file deletions and rewrites on an LTFS volume, the values returned will be estimates that will become less accurate as you write more data to an LTFS volume, delete files, or replace existing files with new versions.

11. **Update the ArGest Backup catalog Info when you add data to an existing tape.** If you are adding new data to a tape that was previously imported into ArGest Backup, the catalog must be updated by reimporting the tape. Because ArGest Backup only knows about LTFS volumes through the state of their catalog info, files added to an existing tape will not be seen in that volume's catalog unless you reimport that volume. The new import will replace the existing catalog.

12. **LTFS doesn't offer a verification mechanism.** Unlike BRU formatted tapes, LTFS volumes can only be verified on the system where they were created and written and then only by performing a full file-by-file comparison against the original files on disk.

13. **LTFS tapes can't be used for normal backup and archival operations**. Once a tape is formatted for use as an LTFS volume, it is currently not possible to use the tape as a normal data tape for ArGest Backup operation. This means that tapes defined for LTFS use can only be used for LTFS purposes. If you decide to return an LTFS formatted tape to normal data operations, you must be using LTFS version 1.2.2 or later which includes the **unltfs** utility for removal of the LTFS format and information from a tape.

ArGest Backup License Agreement

Other World Computing, Inc.
LICENSE AGREEMENT

This License Agreement ("Agreement"), provided by The Other World Computing, Inc. ("TOLIS"), governs the use of the object code version of the BRU brand computer software, documentation and materials accompanying this Agreement or otherwise provided in connection herewith (collectively, "Software"), owned by TOLIS, by the person or entity ("Client") that has agreed with the license terms outlined below. For purposes of this Agreement, Software may be BRU, BRU Server, bruCLONE, BRU LE, ArGest Backup, TOLIS Tape Tools or other licensed BRU application.

****

IF YOU DO NOT AGREE WITH THESE TERMS, YOU MUST NOT INSTALL OR USE THE SOFTWARE.
****

1. LICENSE AND USE RESTRICTIONS.

Subject to all other terms of this Agreement including the payment of any applicable fees, TOLIS hereby grants to Client a non-exclusive, non-transferable license, without the right to grant sublicenses, to use one (1) copy of the Software solely for Client's own, internal purposes on one system. If the Software is BRU-PE (Personal Edition), this license is granted for non-commercial use only (For purposes of this Agreement, noncommercial use means solely for the personal use of the natural person who is the licensee hereunder not utilizing the software license as part of a job while working for a company or organization).

The foregoing license includes the right of Client to make a reasonable number of copies of the computer programs contained in the Software solely for backup and archival purposes; provided, however, that all such copies shall be deemed Software for purposes of this Agreement. The foregoing license shall terminate immediately and without notice for any breach of this Agreement by Client, including any failure to pay fees when due. Upon any such termination, Client shall immediately destroy or delete any and all Software and promptly confirm in writing that Client has done so.

For bruCLONE, this license is assigned to one specific computer system and may not be used on multiple systems even when not being used on the original system except as outlined in the transferability clause below.

If cloning more than one system, one license is required for each system to be cloned.

Transferability to a new computer system is permitted when the original system upon which a given license is currently installed is to be retired, decommissioned, or otherwise taken out of daily utilization by the originally licensed client. It is permissible to retain use of the licensed software on the original system for a period of overlap required to allow the transfer of data from the original system to the replacement system. Once the license has been transferred to the replacement system and the replacement system is placed in normal use, the license must be removed from the original system. At no other time is it allowable to utilize the same software license on two or more different computers at the same time.

The Software is and shall remain the sole and exclusive confidential and proprietary property of TOLIS, subject to protection under the intellectual property laws of the United States and those throughout the world. Client agrees not to use or disclose the Software, during and after the term of this Agreement, except as expressly permitted by this Agreement. Client further agrees not to modify the Software, remove any notices or markings on the Software, or reverse compile, reverse assemble, reverse engineer or otherwise attempt to learn or disclose the trade secrets contained in the Software, transfer the Software in whole or in part over a network, or permit any third party to do any of the foregoing. Nothing in this Agreement shall be construed as conferring any license under any of TOLIS's intellectual property rights, whether by estoppel, implication, or otherwise, except for those licenses expressly granted herein.

2. WARRANTY AND DISCLAIMER.

TOLIS warrants that for a period of sixty (60) days from the date of receipt by Client of the Software, the media on which the Software was delivered shall be without defects in materials or workmanship. TOLIS agrees to replace any defective media which is returned to TOLIS within the foregoing sixty (60) day period. TOLIS may make available to Client additional services, including updates, enhancements or improvements of or to the Software, under separate written agreement, and for additional payment.

THE FOREGOING WARRANTY IS THE ONLY WARRANTY GIVEN HEREUNDER. EXCEPT AS OTHERWISE PROVIDED ABOVE, THE SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT ANY WARRANTY WHATSOEVER. ALL EXPRESS, IMPLIED OR STATUTORY CONDITIONS, REPRESENTATIONS AND WARRANTIES,

INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED.

Some states do not allow the disclaimer of implied warranties, so the foregoing limitations may not apply to you.

3. LIMITATION OF LIABILITY.

TOLIS SHALL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES ARISING UNDER THIS AGREEMENT OR IN CONNECTION WITH THE SOFTWARE, REGARDLESS OF WHETHER ADVISED BEFOREHAND OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL THE LIABILITY OF TOLIS HEREUNDER EXCEED THE SUM OF ONE HUNDRED DOLLARS ($100), REGARDLESS OF THE CAUSE OF ACTION, IN TORT, CONTRACT OR OTHERWISE.

4. GENERAL.

Any action related to this Agreement shall be governed by the substantive laws of the State of Arizona, without regard to conflicts of law principles. The State and Federal courts located in Maricopa County, Arizona, shall have sole jurisdiction over any dispute arising hereunder, and the parties hereby consent to the personal jurisdiction of such courts. Neither this Agreement, nor any rights hereunder, may be assigned by operation of law or otherwise, in whole in part, by Client without the prior, written permission of TOLIS. Any sale of more than fifty percent (50%) of the common voting stock of, or other right to control, Client shall be deemed an assignment. Any purported assignment without such permission shall be void.

Any waiver of any rights of TOLIS under this Agreement must be in writing, signed by TOLIS, and any such waiver shall not operate as a waiver of any future breach of this Agreement. In the event any portion of this Agreement is found to be illegal or unenforceable, such portion shall be severed from this Agreement, and the remaining terms shall be separately enforced. The parties agree that any breach or threatened breach of this Agreement by Client is likely to cause TOLIS damage that is not fully reparable by payment of damages, and further agree that in such case TOLIS shall be entitled to seek and obtain injunctive or other equitable relief to protect its rights hereunder. Client's performance hereunder and use of the Software shall at all times comply with all applicable laws, rules and regulations, including those governing export of technical information, and Client shall fully indemnify, defend and hold harmless TOLIS against any violation thereof. This Agreement is the

entire agreement between the parties with respect to this subject matter, and supersedes any and all prior or contemporaneous, conflicting or additional communications, negotiations or agreements.

The Other World Computing, Inc.
8687 Via de Venture
Suite 115
Scottsdale, AZ 85258
bruinfo@owc.com