backup backup and restore

doc generated from the script with gendoc bash script, version=3.07

Synopsis

backup [options]

Options:

`-h,--help`	print this help and exit
`-H,--Help`	print full documentation via less and exit
`-V,--version`	print version and exit
`-c,--conf=FILE`	use FILE as the configuration file
`-t,--test`	print intermediate messages for debugging, no **backup** made
`-f,--full`	make a full **backup**
`-n,--named=FILE`	make partial **backup** to named file
`-r,--restore=FILE`	Restore `FILE` in current directory
`-s,--show`	Show a listing of the current backups

Description

backup backups files into zip compressed archives, or restores files from those. Both full and partial backups are possible.

Configuration

The selection of files to be backed up is governed by 6 variables. The first two (`BackupDir` and `DirsToSearch`) tell where backups are stored and where files to be backed up are searched, respectively. The remaining variables are arrays of Bash regular expressions that act on the full pathnames of the file that are **backup** candidates.

Important note: the Bash regular expressions (*ReToSkip below) will be automatically adapted by escaping any dots (.) in them: dots occur frequently and in unescaped form they stand for one character, while in the current context it is more useful to mean a literal dot.

`BackupDir`

is the directory where archives are stored;
For safety reasons you would normally make the
backup directory a mount point for an other disk
than the one(s) containing the information to be
backed up.

`DirsToSearch`

is an array of directories (including any subdirectories) to
be searched.

`DirReToSkip`

is an array of regexps matching directory names to be skipped;
any directory for which the name matches one of
these expressions will not be searched for files to
backup; neither will any of its subdirectories.

`ExtReToSkip`

is an array of regexps matching extensions; files with those
extensions are skipped.

`BaseReToSkip`

is an array of regexps matching basenames to be skipped;
any file for which the basename matches any of

these regexps will not be backed up.

`FileReToSkip`

is an array of regexps matching filenames to be skipped;

Here is an overview of the default settings for these variables:

```
    BackupDir=/backup/$HOSTNAME
 DirsToSearch=(/home /etc)
  DirReToSkip=(t temp tmp ’[Cc]ache’ .thumbnails)
  ExtReToSkip=(auk aux bak bku dep err ’lo[fgt]’ msf old swp tmp toc tok)
 BaseReToSkip=(t)
 FileReToSkip=(~$ ’#$’)
```

In principle, only the last variable (`FileReToSkip`) is sufficient for file selection. For example, the above values can be replaced with the following, equivalent, combination:

```
    BackupDir=/backup/$HOSTNAME
 DirsToSearch=(/home /etc)
  DirReToSkip=()
  ExtReToSkip=()
 BaseReToSkip=()
 FileReToSkip=(
        ’~$ #$’
        ’/t$’
        ’\.(auk|aux|bak|bku|dep|err|lo[fgt]|msf|old|swp|tmp|toc|tok)$’
        ’/(t|temp|tmp|[Cc]ache|.thumbnails)/’
 )
```

Configuration files

The default values for the variables described in the previous section can be changed by means of configuration files. **backup** looks for three configuration files, and uses the first it encounters:

1. a file specified with the `--conf` option;
   if `--conf=-` or `-c-` was specified, no configuration file will be read at all.
2. `~/.backup.conf`, if it exists
3. `/etc/backup.conf`, if it exists

Note that:

- A configuration file containing the default settings shown above would have no effect.
- If you put alternative values for only one or a few variable in your configuration file, the other variables keep their default values.
- If you use the `--test` option, files skipped because of the values in these variables will be reported, together with the values of the variable. Wild cards in those values will then be shown in their expanded form, which may be confusing. So it might be wise to put values with wild cards between quotes. This is why, for example, the ’[Cc]ache’ has been quoted in the defulat settings.
- Values containing parenthesis or `#` characters need quotation anyway.
- There is always more than one way to exclude files from backups; for example, you can exclude files with the `.log` extension by specifying `ExtReToSkip=(log)` or with `BaseReToSkip=(’*.log’)` or with `FileReToSkip=(’\.log$’)`.

Options

`-h,--help`

prints help information and then exits

`-H,--Help`

print full documentation via less and exit

`-V,--version`

prints version and then exits

`-f,--full`

Generates a full backup. Without this option, a partial (incremental) **backup** is made, relative to the most recent **backup** in the **backup** directory i.e. in `/backup/$HOSTNAME` if you did not read a configuration file.

Full backups will obtain filenames `f001.zip`, `f002.zip`, and so on. Similarly, partial backups are named `p001.zip`, `p002.zip`, and so on.

`-c,--conf=FILE`

Use the given FILE as configuration file; Without this option, **backup** looks for /.backup.conf or, if that file does not exist, for etcbackup.conf. If none of these is found, **backup** fails with an error message. See the section ”Configuration files” above.

`-r,--restore`

Don’t back up, but restore from old backups instead. Start **backup** with this option in the directory where the file to be restored existed when it was backed up. You will see a numbered listing of existing backups, from which you can select one by typing its number.

If a file of that name already exists, it will be renamed by adding a `.ver0` extension or, if that exists, `.ver1` or `.ver2` et cetera.

You can select all backups by typing `a`; this results in restoring all files with a datetime string attached to the name.
Or you can type `q` to cancel your restore request.

`-n,--named=NAME`

Name for the new **backup** file (instead of pxxx or fxxx). One application is to use this for an hourly **backup** by adding this entry to your crontab:

```
1 * * * * backup --named=$t0(date +%H)
```

If the name has the format `txxx`, like in the above example, the **backup** will be relative to the most recent of all files `[fpt][0-9][0-9][0-9].zip`; otherwise it will be relative to the most recent of `p[0-9][0-9][0-9].zip`.

`-s,--show`

Shows a listing of current backups with the number of files, size of the **backup** in characters and in human readable form, date, and time.

`-t,--test`

print debugging messages while running. Don’t actually make a backup.

Changes

- added –named option

Author and copyright

`Author`	Wybo Dekker
`Email`	wybo@dekkerdocumenten.nl
`License`	Released under the GNU General Public License

Functions used:

configure

`parameters:` -

`description:` Sets the defaults, and then applies any changes defined in the configuration file if it exists (`/etc/backup.conf` by default)

`globals set:` `BackupDir DirsToSearch DirReToSkip ExtReToSkip BaseReToSkip FileReToSkip conffile`

`globals used:` `BackupDir conffile myname`

`returns:` 0

setselector

`parameters:` -

`description:` Find the last full or (if the `--full` option was not used) partial **backup** file: that file is the reference file: it's mtime will be used, by setting selector to `-newer␣<file>`. If it's a full backup, selector is made empty.
If there aren't any backups yet, the **backup** must be full, or an error message is issued.
Also, sets the name for the new **backup** file to `pnnn.zip` or `fnnn.zip`, where `nnn` is one higher than that of the reference file.

`globals set:` `selector newbackup`

`globals used:` `full`

`returns:` 1 if no **backup** file could be found.

human

`parameters:` 1: a number of bytes

`description:` prints the number in human-readable form

`globals set:` -

`globals used:` -

`returns:` 0

showbackups

`parameters:` -

`description:` show the current backups

`globals set:` -

`globals used:` -

`returns:` 0

backupsof

`parameters:` 1: filename in current directory

`description:` Find basenames, without extension, of all archives containing the given filename.

`globals set:` `backupsfound`

`globals used:` -

`returns:` 0

restore

`parameters:` 1: basename of file in current directory
`description:` Restore the given file in the current directory
`globals  set:` `COLUMNS`
`globals used:` `backupsfound REPLY workdir`
`returns:` 0

updatelistings

`parameters:` -
`description:` Update archive listings where necessary
`globals used:` -
`returns:` 0

list

`parameters:` 1: name of archive to be listed
`description:` Create a listing, sorted by decreasing size, of the contents of the archive. Set the modification time of the archive and its listing to to the most recent regular file in the archive.
`returns:` 0

verbo

`parameters:` *: message
`description:` print the argument(s) to standard error, if `$testing` is true
`globals  set:` -
`globals used:` `t0 testing`
`returns:` 0

verbarr

`parameters:` 1: name of an array variable
`description:` echo the name of the variable plus a colon and the array contents in two columns of width 15 and 50 respectively.
`globals  set:` -
`globals used:` -
`returns:`