Bzip2 mini−HOWTO

Table of Contents

Bzip2 mini−HOWTO

David Fetter, david@fetter.org

v2.00, 22 August 1999

This document tells how to use the new bzip2 compression program. The local copy of the sgml at the current site is here, and the "author−itative" sgml is here.

6.Using bzip2 with emacs

7.Using bzip2 with wu−ftpd

8.Using bzip2 with grep

9.Using bzip2 with Netscape under the X.

10.Using bzip2 to recompress other compression formats

1.Introduction

Bzip2 is a groovy new algorithm for compressing data. It generally makes files that are 60−70% of the size of their gzip'd counterparts.

This document will take you through a few common applications for bzip2.

Future versions of the document will have applications of libbzip2, the bzip2 C library which bzip2's author, Julian Seward has kindly written. The bzip2 manual, which includes low−level information about the library, can be found here.

Future versions of the document may also include a summary of the discussion over whether (and how) bzip2 should be used in the Linux kernel.

1.1 Revision History

v2.00

Changed the Using bzip2 with less section so .tar.bzip2 files can actually be read. Thanks to Nicola Fabiano for the correction.

Updated buzzit utility.

Updated tar information.

v1.92

Updated the Getting bzip2 binaries section, including adding S.u.S.E.'s.

v1.91

Corrected a typo and clarified some shell idioms in the section on using bzip2 with tar. Thanks to Alessandro Rubini for these.

Updated the buzzit tool not to stomp on the original bzip2 archive.

Added bgrep, a zgrep−like tool.

v1.9

Clarified the gcc 2.7.* problem. Thanks to Ulrik Dickow for pointing this out.

Added Leonard Jean−Marc's elegant way to work with tar.

Added Linus Ãkerlund's Swedish translation.

Fixed the wu−ftpd section per Arnaud Launay's suggestion.

Moved translations to their own section.

v1.8

Put buzzit and tar.diff in the sgml where they belong. Fixed punctuation and formatting. Thanks to Arnaud Launay for his help correcting my copy. :−)

Dropped xv project for now due to lack of popular interest.

Added teasers for future versions of the document.

v1.7

Added buzzit utility. Fixed the patch against gnu tar.

v1.6

Added TenThumbs' Netscape enabler.

Also changed lesspipe.sh per his sugestion. It should work better now.

v1.5

Added Arnaud Launay's French translation, and his wu−ftpd file.

v1.4

Added Tetsu Isaji's Japanese translation.

v1.3

Added Ulrik Dickow's .emacs for 19.30 and higher.

(Also corrected jka−compr.el patch for emacs per his suggestion. Oops! Bzip2's doesn't yet(?) have an "append" flag.)

v1.2

Changed patch for emacs so it automagically recognizes .bz2 files.

v1.1

Added patch for emacs.

v1.0

Round 1.

2.Getting bzip2

Bzip2's home page is at The UK home site. The United States mirror site is here.

2.1 Bzip2−HOWTO in your language

French speakers may wish to refer to Arnaud Launay's French documents. The web version is here, and you can use ftp here Arnaud can be contacted by electronic mail at this address

Japanese speakers may wish to refer to Tetsu Isaji's Japanese documents here. Isaji can be reached at his home page, or by electronic mail at this address.

Swedish speakers may wish to refer to Linus Ãkerlund's Swedish documents here. Linus can be reached by electronic mail at this address.

2.2 Getting bzip2 precompiled binaries

See the home sites.

2.3 Getting bzip2 sources

They come from the Official sites (see Getting Bzip2 for where.

2.4 Compiling bzip2 for your machine

If you have gcc 2.7.*, change the line that reads

```
CFLAGS = −O3 −fomit−frame−pointer −funroll−loops
```

to

```
CFLAGS = −O2 −fomit−frame−pointer
```

that is, replace −O3 with −O2 and drop the −funroll−loops. You may also wish to add any −m* flags (like −m486, for example) you use when compiling kernels.

Avoiding −funroll−loops is the most important part, since this will cause many gcc 2.7's to generate wrong code, and all gcc 2.7's to generate slower and larger code. For other compilers (lcc, egcs, gcc 2.8.x) the default CFLAGS are fine.

After that, just `make` it and install it per the README.

3.Using bzip2 by itself

Read the Fine Manual Page :)

4.Using bzip2 with tar

Listed below are three ways to use bzip2 with tar, namely

4.1 Easiest to set up:

This method requires no setup at all. To un−tar the bzip2'd tar archive, foo.tar.bz2 in the current directory, do

```
/path/to/bzip2 −cd foo.tar.bz2 | tar xf −
```

or

```
tar --use-compress-prog=bzip2 xf foo.tar.bz2
```

These work, but can be a PITA to type often.

4.2 Easy to set up, fairly easy to use, no need for root privileges:

Thanks to Leonard Jean−Marc for the tip. Thanks also to Alessandro Rubini for differentiating bash from the csh's.

In your .bashrc, you can put in a line like this:

```
alias btar='tar --use-compress-program /usr/local/bin/bzip2 '
```

In your .tcshrc, or .cshrc, the analogous line looks like this:

```
alias btar 'tar --use-compress-program /usr/local/bin/bzip2'
```

4.3 Also easy to use, but needs root access.

Update your tar to GNU's newest version, which is currently 1.13.10. It can be found at GNU's ftp site or any mirror.

5.Using bzip2 with less

To uncompress bzip2'd files on the fly, i.e. to be able to use "less" on them without first bunzip2'ing them, you can make a lesspipe.sh (man less) like this:

```
#!/bin/sh
# This is a preprocessor for 'less'.  It is used when this environment
# variable is set:   LESSOPEN="|lesspipe.sh %s"

  case "$1" in
  *.tar) tar tvvf $1 2>/dev/null ;; # View contents of various tar'd files
```

```
  *.tgz) tar tzvvf $1 2>/dev/null ;;
# This one work for the unmodified version of tar:
  *.tar.bz2) bzip2 −cd $1 $1 2>/dev/null | tar tvvf − ;;
#This one works with the patched version of tar:
# *.tar.bz2) tyvvf $1 2>/dev/null ;;
  *.tar.gz) tar tzvvf $1 2>/dev/null ;;
  *.tar.Z) tar tzvvf $1 2>/dev/null ;;
  *.tar.z) tar tzvvf $1 2>/dev/null ;;
  *.bz2) bzip2 −dc $1  2>/dev/null ;; # View compressed files correctly
  *.Z) gzip −dc $1  2>/dev/null ;;
  *.z) gzip −dc $1  2>/dev/null ;;
  *.gz) gzip −dc $1  2>/dev/null ;;
  *.zip) unzip −l $1 2>/dev/null ;;
  *.1|*.2|*.3|*.4|*.5|*.6|*.7|*.8|*.9|*.n|*.man) FILE=`file −L $1` ; # groff src
    FILE=`echo $FILE | cut −d ' ' −f 2`
    if [ "$FILE" = "troff" ]; then
      groff −s −p −t −e −Tascii −mandoc $1
    fi ;;
  *) cat $1 2>/dev/null ;;
#  *) FILE=`file −L $1` ; # Check to see if binary, if so −− view with 'strings'
#    FILE1=`echo $FILE | cut −d ' ' −f 2`
#    FILE2=`echo $FILE | cut −d ' ' −f 3`
#    if [ "$FILE1" = "Linux/i386" −o "$FILE2" = "Linux/i386" \
#         −o "$FILE1" = "ELF" −o "$FILE2" = "ELF" ]; then
#      strings $1
#    fi ;;
  esac
```

6.Using bzip2 with emacs

6.1 Changing emacs for everyone:

I've written the following patch to jka−compr.el which adds bzip2 to auto−compression−mode.

Disclaimer: I have only tested this with emacs−20.2, but have no reason to believe that a similar approach won't work with other versions.

To use it,

1. Go to the emacs−20.2/lisp source directory (wherever you untarred it)
2. Put the patch below in a file called jka−compr.el.diff (it should be alone in that file ;).
3. Do

   ```
   patch < jka−compr.el.diff
   ```

4. Start emacs, and do

   ```
   M−x byte−compile−file jka−compr.el
   ```

5. Leave emacs.
6. Move your original jka−compr.elc to a safe place in case of bugs.
7. Replace it with the new jka−compr.elc.
8. Have fun!

```
--- jka-compr.el        Sat Jul 26 17:02:39 1997
+++ jka-compr.el.new    Thu Feb  5 17:44:35 1998
@@ -44,7 +44,7 @@
 ;; The variable, jka-compr-compression-info-list can be used to
 ;; customize jka-compr to work with other compression programs.
 ;; The default value of this variable allows jka-compr to work with
-;; Unix compress and gzip.
+;; Unix compress and gzip.  David Fetter added bzip2 support :)
 ;;
 ;; If you are concerned about the stderr output of gzip and other
 ;; compression/decompression programs showing up in your buffers, you
@@ -121,7 +121,9 @@


 ;;; I have this defined so that .Z files are assumed to be in unix
-;;; compress format; and .gz files, in gzip format.
+;;; compress format; and .gz files, in gzip format, and .bz2 files,
+;;; in the snappy new bzip2 format from http://www.muraroa.demon.co.uk.
+;;; Keep up the good work, people!
 (defcustom jka-compr-compression-info-list
   ;;[regexp
   ;; compr-message  compr-prog  compr-args
@@ -131,6 +133,10 @@
      "compressing"    "compress"     ("-c")
      "uncompressing"  "uncompress"   ("-c")
      nil t]
+    ["\\.bz2\\'"
+     "bzip2ing"        "bzip2"         ("")
+     "bunzip2ing"      "bzip2"         ("-d")
+     nil t]
     ["\\.tgz\\'"
      "zipping"        "gzip"         ("-c" "-q")
      "unzipping"      "gzip"         ("-c" "-q" "-d")
```

6.2 Changing emacs for one person:

Thanks for this one go to Ulrik Dickow, ukd@kampsax.dk, Systems Programmer at Kampsax Technology:

To make it so you can use bzip2 automatically when you aren't the sysadmin, just add the following to your .emacs file.

```
;; Automatic (un)compression on loading/saving files (gzip(1) and similar)
;; We start it in the off state, so that bzip2(1) support can be added.
;; Code thrown together by Ulrik Dickow for ~/.emacs with Emacs 19.34.
;; Should work with many older and newer Emacsen too.  No warranty though.
;;
(if (fboundp 'auto-compression-mode) ; Emacs 19.30+
    (auto-compression-mode 0)
  (require 'jka-compr)
  (toggle-auto-compression 0))
;; Now add bzip2 support and turn auto compression back on.
(add-to-list 'jka-compr-compression-info-list
```

```
             ["\\.bz2\\(~\\|\\.~[0-9]+~\\)?\\'"
              "zipping"        "bzip2"         ()
              "unzipping"      "bzip2"         ("-d")
              nil t])
(toggle-auto-compression 1 t)
```

7.Using bzip2 with wu−ftpd

Thanks to Arnaud Launay for this bandwidth saver. The following should go in /etc/ftpconversions to do on−the−fly compressions and decompressions with bzip2. Make sure that the paths (like /bin/compress) are right.

```
 :.Z:  :  :/bin/compress -d -c %s:T_REG|T_ASCII:O_UNCOMPRESS:UNCOMPRESS
 :   : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
 :.gz: :  :/bin/gzip -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:GUNZIP
 :   : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP
 :.bz2: :  :/bin/bzip2 -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:BUNZIP2
 :   : :.bz2:/bin/bzip2 -9 -c %s:T_REG:O_COMPRESS:BZIP2
 :   : :.tar:/bin/tar -c -f - %s:T_REG|T_DIR:O_TAR:TAR
 :   : :.tar.Z:/bin/tar -c -Z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+COMPRESS
 :   : :.tar.gz:/bin/tar -c -z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+GZIP
 :   : :.tar.bz2:/bin/tar -c -y -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+BZIP2
```

8.Using bzip2 with grep

The following utility, which I call bgrep, is a slight modification of the zgrep which comes with Linux. You can use it to grep through files without bunzip2'ing them first.

```
#!/bin/sh

# bgrep -- a wrapper around a grep program that decompresses files as needed
PATH="/usr/bin:$PATH"; export PATH

prog=`echo $0 | sed 's|.*/||'`
case "$prog" in
        *egrep) grep=${EGREP-egrep}     ;;
        *fgrep) grep=${FGREP-fgrep}     ;;
        *)      grep=${GREP-grep}       ;;
esac
pat=""
while test $# -ne 0; do
  case "$1" in
  -e | -f) opt="$opt $1"; shift; pat="$1"
           if test "$grep" = grep; then  # grep is buggy with -e on SVR4
             grep=egrep
           fi;;
  -*)      opt="$opt $1";;
   *)      if test -z "$pat"; then
             pat="$1"
```

```
           else
             break;
           fi;;
  esac
  shift
done

if test −z "$pat"; then
  echo "grep through bzip2 files"
  echo "usage: $prog [grep_options] pattern [files]"
  exit 1
fi

list=0
silent=0
op=`echo "$opt" | sed −e 's/ //g' −e 's/−//g'`
case "$op" in
  *l*) list=1
esac
case "$op" in
  *h*) silent=1
esac

if test $# −eq 0; then
  bzip2 −cd | $grep $opt "$pat"
  exit $?
fi

res=0
for i do
  if test $list −eq 1; then
    bzip2 −cdfq "$i" | $grep $opt "$pat" > /dev/null && echo $i
    r=$?
  elif test $# −eq 1 −o $silent −eq 1; then
    bzip2 −cd "$i" | $grep $opt "$pat"
    r=$?
  else
    bzip2 −cd "$i" | $grep $opt "$pat" | sed "s|^|${i}:|"
    r=$?
  fi
  test "$r" −ne 0 && res="$r"
done
exit $res
```

9.Using bzip2 with Netscape under the X.

tenthumbs@cybernex.net says:

```
I also found a way to get Linux Netscape to use bzip2 for
Content−Encoding just as it uses gzip. Add this to
$HOME/.Xdefaults or $HOME/.Xresources
```

> I use the −s option because I would rather trade some decompressing speed for RAM usage. You can leave the option out if you want to.

```
Netscape*encodingFilters:      \
        x-compress :  : .Z     : uncompress -c  \n\
        compress   :  : .Z     : uncompress -c  \n\
        x-gzip     :  : .z,.gz : gzip -cdq      \n\
        gzip       :  : .z,.gz : gzip -cdq      \n\
        x-bzip2    :  : .bz2   : bzip2 -ds \n
```

10.Using bzip2 to recompress other compression formats

The following perl program takes files compressed in other formats (.tar.gz, .tgz. .tar.Z, and .Z for this iteration) and repacks them for better compression. The perl source has all kinds of neat documentation on what it does and how it does what it does. This latest version takes files as input on the command line. Without command line arguments, it tries to repack every file in the current working directory.

```
#!/usr/bin/perl -w

#######################################################
#                                                     #
# This program takes compressed and gzipped programs  #
# in the current directory and turns them into bzip2  #
# format.  It handles the .tgz extension in a         #
# reasonable way, producing a .tar.bz2 file.          #
#                                                     #
#######################################################
$counter = 0;
$saved_bytes = 0;
$totals_file = '/tmp/machine_bzip2_total';
$machine_bzip2_total = 0;

@raw = (defined @ARGV)?@ARGV:<*>;

foreach(@raw) {
    next if /^bzip/;
    next unless /\.(tgz|gz|Z)$/;
    push @files, $_;
}
$total = scalar(@files);

foreach (@files) {
    if (/tgz$/) {
        ($new=$_) =~ s/tgz$/tar.bz2/;
    } else {
        ($new=$_) =~ s/\.g?z$/.bz2/i;
    }
    $orig_size = (stat $_)[7];
    ++$counter;
    print "Repacking $_ ($counter/$total)...\n";
    if ((system "gzip -cd $_ |bzip2 >$new") == 0) {
        $new_size = (stat $new)[7];
        $factor = int(100*$new_size/$orig_size+.5);
        $saved_bytes += $orig_size-$new_size;
        print "$new is about $factor% of the size of $_. :",($factor<100)?')':'(',"\n";
        unlink $_;
    } else {
        print "Arrgghh!  Something happened to $_: $!\n";
```

```
    }
}
print "You've "
    , ($saved_bytes>=0)?"saved ":"lost "
    , abs($saved_bytes)
    , " bytes of storage space :"
    , ($saved_bytes>=0)?")":"("
    , "\n"
    ;

unless (-e '/tmp/machine_bzip2_total') {
    system ('echo "0" >/tmp/machine_bzip2_total');
    system ('chmod', '0666', '/tmp/machine_bzip2_total');
}


chomp($machine_bzip2_total = `cat $totals_file`);
open TOTAL, ">$totals_file"
     or die "Can't open system-wide total: $!";
$machine_bzip2_total += $saved_bytes;
print TOTAL $machine_bzip2_total;
close TOTAL;

print "That's a machine-wide total of ",`cat $totals_file`," bytes saved.\n";
```
