

– General Linux 2 –

Make and install programs from source

[5]

(Linux Professional Institute Certification)

a

```
.~.  
/V\      by: geoffrey robertson  
//  \\      geoffrey@zip.com.au  
@.__.@
```

\$Id: gl2.102.3.slides.tex,v 1.3 2003/05/30 06:03:32 waratah Exp \$

^aCopyright © 2002 Geoffrey Robertson. Permission is granted to make and distribute verbatim copies or modified versions of this document provided that this copyright notice and this permission notice are preserved on all copies under the terms of the GNU General Public License as published by the Free Software Foundation—either version 2 of the License or (at your option) any later version.

List of Slides

(2.2) 102 Installation & Package Mgt. [24]

2.102.1 Design hard disk layout [2]

2.102.2 Install a boot manager [3]

2.102.3 Make and install programs from source [5]

2.102.4 Manage shared libraries [3]

2.102.5 Use Debian package management [5]

2.102.6 Use Red Hat Package Manager (RPM) [6]

Make and install programs from source [5]

Objective

Candidates should be able to build and install an executable program from source. This objective includes being able to unpack a file of sources.

Candidates should be able to make simple customisations to the Makefile, for example changing paths or adding extra include directories.

Make and install programs from source [5]

Key files, terms, and utilities

gunzip

gzip

bzip2

tar

configure

make

Make and install programs from source [5]

Resources of interest

LPI Linux Certification in a Nutshell

by Jeffrey Dean

O'Reilly

LPIC 1 Certification Bible

Angie Nash and Jason Nash

Hungry Minds

Source Code Distribution

To distribute software in the form of source code a **source tree** is archived into one file using the tar command and then compressed. The resulting file is called a **tarball**.

Source code may also be distributed using the package management tools of a particular distribution.

Debian `apt-get install kernel-source-2.2.27`

Redhat `rpm -Uhv at-3.1.8-23.src.rpm`

Tarball `tdb-1.0.6.tar.gz`

Installing the trivial database tdb

Download

Locate and download the tarball

- googling for it: <http://google.com>
- search on freshmeat: <http://freshmeat.net>
- see if it lives on sourceforge: <http://www.sf.net>

Download the tarball to a suitable directory such as `/tmp`.

Installing the trivial database tdb

Unpack

The tarball file is a compressed archived source tree.

Most commonly the file will be compressed using either `gzip` or `bzip2`

GNU `tar` can uncompress and unpack the archive:

```
$ tar zxvf tdb-1.0.6.tar.gz ↵
```

or

```
$ tar jxvf tdb-1.0.6.tar.bz2 ↵
```

Installing the trivial database tdb

cd into the tree

The unpacked tarball creates a source tree. The base of which is the name of the program

```
$ ls ↵
```

```
tdb-1.0.6  tdb-1.0.6.tar.gz
```

```
$ cd tdb-1.0.6 ↵
```

```
$ ls ↵
```

```
configure  tdb.c  tdb.h  README  INSTALL  COPYING
```

```
...
```

Installing the trivial database tdb

cd into the tree

```
$ ls -w 70 ↵
```

```
acconfig.h      install-sh      stamp-h.in     tdb.h
aclocal.m4      ltconfig       tdb.3          tdbiterate.c
AUTHORS         ltmain.sh      tdb.c          tdb_open.3
ChangeLog       Makefile.am    tdb_chainlock.3 tdb.spec
config.guess    Makefile.in    tdb_close.3   tdbspeed.c
config.h.in     missing       tdb_delete.3  tdb_store.3
config.sub      mkinstalldirs tdbdump.c     tdbtest.c
configure       NEWS          tdb_error.3   tdbtool.c
configure.in    README        tdb_exists.3  tdbtorture.c
COPYING         spinlock.c     tdb_fetch.3   tdb_traverse.3
INSTALL         spinlock.h     tdb_firstkey.3 TODO
```

Installing the trivial database tdb

```
./configure
```

```
$ file configure ↵
```

```
configure: Bourne shell script text executable
```

```
$ head -5 configure ↵
```

```
#!/bin/sh
```

```
# Guess values for system-dependent variables
```

```
# Create Makefiles.
```

```
# Generated automatically using autoconf version 2.13
```

Installing the trivial database tdb

`./configure`

```
$ ./configure ↵
  creating cache ./config.cache
checking for a BSD compat install... /usr/bin/install -c
checking whether build environment is sane... yes
checking whether make sets $MAKE... yes
checking for working aclocal... found
...
creating ./config.status
creating Makefile
creating config.h
```

Installing the trivial database tdb

The Makefile

```
SHELL = /bin/sh
CC = gcc
CFLAGS = -g -O2
prefix = /usr/local
includedir = $prefix/include
...
tdbtool: $(tdbtool_OBJECTS) $(tdbtool_DEPENDENCIES)
        @rm -f tdbtool
        $(LINK) $(tdbtool_LDFLAGS) $(tdbtool_OBJECTS)
...
distclean: distclean-am
        -rm -f config.status
```

Installing the trivial database tdb

make

```
$ make ↵  
/bin/sh ./libtool --mode=compile gcc -DHAVE_CONFIG_H -I.  
-I. -I. -g -O2 -c tdb.c  
mkdir .libs  
gcc -DHAVE_CONFIG_H -I. -I. -I. -g -O2 -c -fPIC -DPIC  
tdb.c -o .libs/tdb.lo  
gcc -DHAVE_CONFIG_H -I. -I. -I. -g -O2 -c tdb.c -o tdb.o  
>/dev/null 2>&1  
mv -f .libs/tdb.lo tdb.lo  
/bin/sh ./libtool --mode=compile gcc -DHAVE_CONFIG_H -I.  
-I. -I. -g -O2 -c spinlock.c  
...
```

Installing the trivial database tdb

```
make install
```

```
su -c 'make install'
Password:
make[1]: Entering directory `/tmp/tdb-1.0.6'
/bin/sh ./mkinstalldirs /usr/local/lib
/bin/sh ./libtool --mode=install /usr/bin/install -c
libtdb.la /usr/local/lib/libtdb.la
...
chmod 644 /usr/local/lib/libtdb.a
PATH="$PATH:/sbin" ldconfig -n /usr/local/lib
```

Using tdb

```
$ tdbtool ←
```

```
tdb> ?
```

```
tdbtool:
```

```
  create  dbname      : create a database
  open    dbname      : open an existing database
  erase                    : erase the database
  dump    dumpname    : dump the database as strings
  insert  key  data    : insert a record
  store   key  data    : store a record (replace)
  show    key          : show a record by key
  delete  key          : delete a record by key
  list                    : print the database hash table and freelist
  free                    : print the database freelist
  1 | first             : print the first record
  n | next              : print the next record
  q | quit              : terminate
  \n                    : repeat 'next' command
tdb>
```

Using tdb

```
$ tdbtool ↵
```

```
tdb> create test.tdb
```

```
tdb> insert 1 thing
```

```
tdb> insert 2 foo
```

```
tdb> insert 3 bar
```

```
tdb> insert 55 whizz
```

```
tdb> show 3
```

```
key 2 bytes
```

```
3
```

```
data 4 bytes
```

```
[000] 62 61 72 00      bar
```

Summary

- `$ tar zxvf my-progy.tar.gz ↵`

Summary

- `$ tar zxvf my-progy.tar.gz ↵`
- `$ cd my-progy ↵`

Summary

- `$ tar zxvf my-progy.tar.gz ↵`
- `$ cd my-progy ↵`
- `$ less README INSTALL ↵`

Summary

- `$ tar zxvf my-progy.tar.gz ↵`
- `$ cd my-progy ↵`
- `$ less README INSTALL ↵`
- `$./configure ↵`

Summary

- `$ tar zxvf my-progy.tar.gz ↵`
- `$ cd my-progy ↵`
- `$ less README INSTALL ↵`
- `$./configure ↵`
- `$ make ↵`

Summary

- `$ tar zxvf my-progy.tar.gz ↵`
- `$ cd my-progy ↵`
- `$ less README INSTALL ↵`
- `$./configure ↵`
- `$ make ↵`
- `$ su -c 'make install' ↵`

Summary

- `$ tar zxvf my-progy.tar.gz ↵`
- `$ cd my-progy ↵`
- `$ less README INSTALL ↵`
- `$./configure ↵`
- `$ make ↵`
- `$ su -c 'make install' ↵`
- `$ my-progy ↵`

The End