NAME

dds2tar − tool for fast tape access

SYNOPSIS

dds2tar [**-f** *device*] [**-t** *indexfile*] [options] *string ...*

DESCRIPTION

dds2tar uses an index to find the files with record seek (a fast operation of DAT devices). Since the file structure of the tape archives is used to extract the files, the archive has to be created by **tar**, compressed only by (the transparent signal processor of) the device. So you can step through the archive very quickly and extract files. The index may be created using **dds2index** or **tar**-vRt and is normally stored as a file on your hard disk.

A tar archive is a sequence of blocks (e.g. 10240 bytes by default), each containing the same number (20 by default) of records, 512 byte each. **dds2tar** reads the tape and writes the tar records of the specified files (that means the header record and the data records of each selected file) to stdout. You may pipe the **dds2tar** output to the stdin of *tar -xvvf -* to restore the files to your disk. (See **EXAMPLES** below.) Before a file is extracted, the records of parent directories of the file are also written to stdout.

The index of the archive should contain enough information to compute the number of the block containing the header of each selected file. **dds2index** will give such a table, **tar -Rvt** e.g. will not (only record numbers are listed). A patch for **GNU tar-1.12** is available, adding the option **--record-file.** Alternatively there are some tricks to get the missing information.

The strings are regular expressions to select the files. The matching algorithm is the one from GNU tar. If the option **-l** is given, the matched file names are printed to stdout (You may not pipe this list of pathnames to tar!).

The default device is */dev/rmt0*, which may be overridden with the environment variable **TAPE**, which in turn may be overridden with the **-f** *device* option. The device must be a SCSI tape device.

OPTIONS

-f *devicefile* Device of the tape archive. Must be a SCSI tape device.

-t *indexfile*
: Specifies the index file (default is stdin).

-s #
: Set the number of the first tape block of the archive. This option is useful only if the index file contains the verbose output of **tar -Rvt.** Any information about the first block inside the index file will be overridden by this option. If no information is available, the archive has to be the first file of the tape. If you have positioned your tape at the first block of the archive, you can use

dds2tar ‘mt-dds‘ -t index ... | tar -f - ...

to complete the information of the output of **tar -Rvt** stored in the index file.

-b #
: Set the blocksize of the archive (tar -b #). This option is useful only if the index file contains the verbose output of tar (or if you have problems with the size of the internal buffer of dds2tar). Any information about the blocksize inside the index file will be overridden by this option. If no information is available, the default blocksize of tar is used.

-z
: The index file should be read and stored in compressed mode.

OPTIONS you didn’t really need

--z, --no-compress
: Don’t filter the archive file through gzip.

-q, --quick
: Don’t extract the parent directories of the selected objects from tape. **--body** Write only the first selected file to stdout. This is useful if you want to read a file or extract an archive which is part of the current archive.

-v,**--verbose**
verbose mode.

--hash-mode
Print a hash sign for each MB.

-V,**--version**
Print only the Version Number to stderr.

-l Don't access the tape but print the file names to stdout. You may not pipe this list of pathnames into tar.

--extract
The stdout is closed and opened by a pipe to the command **tar -fxb - 1 .** You may find this option convenient, I like to pipe the output to tar by hand.

EXAMPLES

Example of **getting the index** from the default tape /dev/rmt0 and storing it in file archive.idx:

```
dds2index -t archive.idx
```

Alternatively you can use a patched version of tar to create an index file. With the patch you can direct the errors and warning to stdout and the index information including information about the blocksize and the number of the first block to a file:

```
tar -t --record-file archive.idx
```

If the archive is the first file of the tape and the blocksize is the default of 20, you can use the verbose output of tar (-Rv) as an index file.

```
tar -t -v -R | tee archive.idx
```

If the archive is not the first file of the tape, you can store all the necessary information inside the index file with the use of **mt-dds** and **tar** :

```
mt asf ...
mt-dds tell > archive.idx
tar -tvR >>archive.idx
```

Example of **using dds2tar** to extract the gnu library (all files containing the string "glibc" in filename) from the default tape /dev/rmt0, using the previously stored index file archive.idx:

```
dds2tar -t archive.idx '*glibc*' | tar xvvf -
```

To see in advance what would happen in the previous command without actually writing anything to your disk, you may use:

```
dds2tar -t archive.idx '*glibc*' | tar tvvf -
```

Example of checking the matches. You may try:

```
dds2tar -t archive.idx -l '*glibc*'
```

BACKGROUND INFORMATION

tapes

A tape device handles all I/O (read, write, seek) in units of *tape records*. The bigger a tape record, the more effective usually is the access (and the less gaps are on QIC-tapes). However, normally a program will only read or write complete tape records.

Normal tape drives allow to seek only relative to the current position. However, some newer SCSI-2 tapes, i.e. DAT, conforming to the DDS standard, keep track of the absolute position on the tape by inserting the tape record number inside each track. This number can be read while the fast seek is performed.

The **tar(1)** program uses a slightly different terminology. It calls *tape blocks* what normally is called *tape records*. In the following sections we use the tar terminology to avoid confusion.

tar

The unit inside a **tar** archive is a *tar record* with a fixed length of 512 bytes. Every file, directory or soft link will occupy at least one tar record of information about pathname, permission information and so on called header record. The data of each file is stored in additional tar records directly after the header record of that file.

tar reports the *tar record number* of every header record in the archive with its -R option. tar counts the records continuously, starting with **0** (if invoked as tar -tR) or with **1** (if invoked as tar -cR).

tar handles multiple records as a *tar block*, mainly to make the access of tapes (or disks) more efficient (and save tape space of QIC-tapes). tar only writes and reads full blocks to or from an archive. The -b option of tar controls, how many records are in one block. The default number of records per block is **20**. This number is usually called the *tar block size*. However, this term is a little bit confusing, since it does not mean the number of bytes in a block. Thus a perhaps better name would be the *tar blocking factor*.

tar on tapes

tar writes or reads its archive to or from tape in units of tar blocks. As stated above, only a complete tape block may be transferred to/from tape. To extract a specific tar block from tape, one has to read an entire tape block into a buffer and extract the specified tar record from the buffer manually. If you would like to read a tar record with a given number, you have to know the number of the first tape block of the archive and the tar block size to compute the number of the tape block witch contains the tar record to read. If the tar archive is the first file on the tape, the *tape block number* is the equal to the *tar block number*.

Example: A file with the tar record number 1234 (records start with 0) may be found in a tape tar archive, written with a blocking factor of 20. It may be found in the tar block with the number

blk = (int) 1234/20 = (int) 61.7 = 61

which is also the tape block number. The requested file is within this tar block at the record offset

rec = 1234-(61*20) = 14

in 512 byte units.

If a current archive is not the first archive on the tape, then the number of *tape blocks* of all previous archives has to be added to the block number computed above, to get the *current tape block number*. The number of previous tape records should be obtained from DDS devices when the tape is positioned at the beginning of the current archive (use **mt-dds** without arguments for example).

Example: Assuming the archive in the above example to be the second file on a tape, and the archive starts at tape block 20222. Then we will find our file with tar record number 1234 in the tape block

tblk = 20222 + (int) 1234/20 = 20283

on the tape. The record offset inside the tape block will be the same as above.

WARNING

This program can only read records (tar is calling them tape blocks) up to 32 kbytes due to the limitations of the Linux device driver. The extracted archive is written to stdout with a block size of 512 bytes.

ENVIRONMENT

The environment variable **TAPE** overrides the default tape device /dev/rmt0. The variable **DDS2TAR cat be used to give some options, e.g. --compress, -z, -s # , -b #.**

SEE ALSO

dds2index(1), mt(1), mt-dds(1), tar(1)

HISTORY

This program was created to use the fast seek operation of my DAT streamer. The tapes are called dds (digital data storage). Since the program will write a tar archive to stdout, I called this program **dds2tar**. If I created the index file, I'm now able to restore a file of 1MB within one minute even if the tape contains more than 2GB of data.

Thanks to Andreas (Andreas_Bagge@h2.maus.de), who has written a nice manual page for the overloaded version 1.1.3 of the program dds2tar (I added too much features ...) His manual page for dds2tar-1.1.3 gave me the idea how to split the program dds2tar into the peaces dds2tar, dds2index and mt-dds. Additionally his manual page was the starting point for this page.

Since the version 2.2 has a very robust algorithm to read the index file and the ability of pattern matching, a lot of options where obsolete and has been deleted. I tried to make dds2tar as simple I can.

AUTHOR

J"org Weule (weule@cs.uni-duesseldorf.de), Phone +49 211 751409. This software is available at ftp.uni-duesseldorf.de:/pub/unix/apollo and sunsite.unc.edu:/pub/Linux/system/Backup