# CS360 Lab #6 - Tarx

- James S. Plank
- CS360
- Lab written in February, 2019.
- Lab directory: **/home/jplank/cs360/labs/Lab-6-Tarx**
- Lab writeup: http://web.eecs.utk.edu/~jplank/plank/classes/cs360/360/labs/Lab-6-Tarx/index.html
- There is a makefile in **makefile**. It assumes that **tarx.c** is in the **src** directory.

---

## Description of tarx

**Tarx** recreates a directory from a tarfile that was created with **tarc**, including all file and directory protections and modification times. It reads the tarfile from standard input.

That's really all of the description **tarx** needs. But I'll walk you through an example. Take a look at the directory **test-dir**, in the lab directory:

```
UNIX> ls -li /home/jplank/cs360/labs/Lab-6-Tarx | grep test-dir
 13270543 drwxr-xr-x. 4 jplank jplank    84 Dec 25  2009 test-dir
UNIX> ( cd /home/jplank/cs360/labs/Lab-6-Tarx ; ls -li test-dir )
total 12
134248998 -r-x-wxr--. 2 jplank jplank 30 Apr  8  2009 lTFYSC
278215456 drwx--xr-x. 2 jplank jplank 76 Aug 23  2009 Md1jUzS6
     7980 -rw----r-x. 1 jplank jplank  3 Feb 12  2009 sOdKRmzyPT3mXX4
134248994 dr-xr--r-x. 2 jplank jplank 83 Jul  5  2009 T7ZdQl
 13271404 -r-x-w-r--. 1 jplank jplank 32 Sep 29  2009 TkB
UNIX> ( cd /home/jplank/cs360/labs/Lab-6-Tarx ; ls -li test-dir/Md1jUzS6 )
total 16
278178660 -rwxr--r-x. 1 jplank jplank 78 Oct 16  2009 aA4X7tu6u
268648073 -r--r--r--. 1 jplank jplank 17 Sep  7  2009 QFIoNAT
134248996 -r--rw-r--. 2 jplank jplank  8 Apr 18  2009 tBSlghe
278176150 -r-----r--. 1 jplank jplank 71 Jul 14  2009 ymSZrTV0u0vrAHL
UNIX> ( cd /home/jplank/cs360/labs/Lab-6-Tarx ; ls -li test-dir/T7ZdQl )
total 16
134248996 -r--rw-r--. 2 jplank jplank  8 Apr 18  2009 aDH
134248997 -r-x--xr-x. 1 jplank jplank 53 May 20  2009 i4obCmgB1ZHpDy7q
134248995 -r---wxr--. 1 jplank jplank 32 Oct 10  2009 JJbBXDSpYO23pb6-
134248998 -r-x-wxr--. 2 jplank jplank 30 Apr  8  2009 RCPdAWLBsz
UNIX>
```

This is a challenging directory from a **tarx** perspective. But first, let's go ahead and create a **tarc** file, and call **tarf** on it to verify our understanding of it. Do this in a fresh directory:

```
UNIX> ls
UNIX> /home/jplank/cs360/labs/Lab-5-Tarc/bin/tarc /home/jplank/cs360/labs/Lab-6-Tarx/test-dir > td.tarc
UNIX> xxd -g 1 td.tarc | head -n 2
0000000: 08 00 00 00 74 65 73 74 2d 64 69 72 0f 7e ca 00  ....test-dir.~..
0000010: 00 00 00 00 ed 41 00 00 95 8b 35 4b 00 00 00 00  .....A....5K....
UNIX> /home/jplank/cs360/labs/Lab-5-Tarc/bin/tarf < td.tarc | head -n 15
Inode 13270543 Mode 40755 Hash 00000000 Mtime 1261800341
Name: test-dir

Inode 278215456 Mode 40715 Hash 00000000 Mtime 1251081491
Name: test-dir/Md1jUzS6

Inode 268648073 Mode 100444 Hash 9802f4b8 Mtime 1252360768
Name: test-dir/Md1jUzS6/QFIoNAT

Inode 278178660 Mode 100745 Hash 2a0b3a18 Mtime 1255692354
Name: test-dir/Md1jUzS6/aA4X7tu6u
```

```
Inode 134248996 Mode 100464 Hash d629e03e Mtime 1240070091
Name: test-dir/Md1jUzS6/tBSlghe
Name: test-dir/T7ZdQl/aDH
UNIX>
```

As you can see, the files **test-dir/Md1jUzS6/tBSlghe** and **test-dir/T7ZdQl/aDH** are hard links to each other. Also, to repeat myself, the bytes in the files are random, so don't try to **cat** them. Instead, use **xxd**:

```
UNIX> xxd -g 1 /home/jplank/cs360/labs/Lab-6-Tarx/test-dir/lTFYSC
0000000: d3 44 27 cf a1 b6 06 4b ca 03 e2 fc b8 2f 34 67  .D'....K...../4g
0000010: a8 4c c6 9e 94 5e 3e 0d 6c 80 d6 99 e1 92        .L...^>.l.....
UNIX>
```

What makes this directory a challenge is the subdirectory **test-dir/T7ZdQl**. You'll note that its write protection is not set. For that reason, you cannot set its mode until after you have created all of its files and subdirectories. That's not a difficult matter, so long as you pay attention to it.

I'll call my **bin/tarx** on the tarfile, and you'll see that it recreates the files and directories with the same protection modes and modification times:

```
UNIX> /home/jplank/cs360/labs/Lab-6-Tarx/bin/tarx < td.tarc
UNIX> ls -li
total 4
    9120 -rw-r--r--. 1 jplank jplank 1018 Feb 22 14:00 td.tarc
13270541 drwxr-xr-x. 4 jplank jplank   84 Dec 25  2009 test-dir
UNIX> ls -li test-dir
total 12
 13259586 -r-x-wxr--. 2 jplank jplank 30 Apr  8  2009 lTFYSC
268648074 drwx--xr-x. 2 jplank jplank 76 Aug 23  2009 Md1jUzS6
 13259588 -rw----r-x. 1 jplank jplank  3 Feb 12  2009 sOdKRmzyPT3mXX4
134249000 dr-xr--r-x. 2 jplank jplank 83 Jul  5  2009 T7ZdQl
     7981 -r-x-w-r--. 1 jplank jplank 32 Sep 29  2009 TkB
UNIX> ls -li test-dir/Md1jUzS6
total 16
268648076 -rwxr--r-x. 1 jplank jplank 78 Oct 16  2009 aA4X7tu6u
268648075 -r--r--r--. 1 jplank jplank 17 Sep  7  2009 QFIoNAT
134249002 -r--rw-r--. 2 jplank jplank  8 Apr 18  2009 tBSlghe
268648077 -r-----r--. 1 jplank jplank 71 Jul 14  2009 ymSZrTV0u0vrAHL
UNIX> ls -li test-dir/T7ZdQl
total 16
134249002 -r--rw-r--. 2 jplank jplank  8 Apr 18  2009 aDH
134249003 -r-x--xr-x. 1 jplank jplank 53 May 20  2009 i4obCmgB1ZHpDy7q
134249001 -r---wxr--. 1 jplank jplank 32 Oct 10  2009 JJbBXDSpYO23pb6-
 13259586 -r-x-wxr--. 2 jplank jplank 30 Apr  8  2009 RCPdAWLBsz
UNIX> xxd -g 1 test-dir/lTFYSC
0000000: d3 44 27 cf a1 b6 06 4b ca 03 e2 fc b8 2f 34 67  .D'....K...../4g
0000010: a8 4c c6 9e 94 5e 3e 0d 6c 80 d6 99 e1 92        .L...^>.l.....
UNIX>
```

---

## The Gradescript

The gradescript goes through the following steps for gradescript *i*.

- It calls your **bin/tarx** on the file **Gradescript-Examples/*i*.tarc**. That should create the directory *i* in the current directory. It also puts stdout and stderr into **tmp-*i*-your-stdout.txt**. and **tmp-*i*-your-stderr.txt**.
- It limits the number of file descriptors to 10 in the **bin/tarx** call.
- It calls **/home/jplank/cs360/labs/Lab-5-Tarc/bin/tarf** on **Gradescript-Examples/*i*.tarc**, stripping out the inodes from stdout. It puts stdout and stderr into **tmp-*i*-correct-stdout.txt**. and **tmp-*i*-correct-stderr.txt**.
- If one of them flagged an error and the other one didn't (which is checked by seeing if either program wrote into stderr), then it's an error.
- If both flagged errors, then your program is correct, and the gradescript exits.
- Otherwise, it calls **/home/jplank/cs360/labs/Lab-5-Tarc/bin/tarc** on your new directory *i* and flags errors if there are any.

- If no errors, then it calls **/home/jplank/cs360/labs/Lab-5-Tarc/bin/tarc** on your new directory *i* and pipes the output to **/home/jplank/cs360/labs/Lab-5-Tarc/bin/tarf**, stripping out the inodes, and putting the output into **tmp-*i*-tarf-output.txt**.
- It tests to see that **tmp-*i*-correct-stdout.txt** are the same **tmp-*i*-tarf-output.txt**.

You should note that the tarfiles may or may not be created by **/home/jplank/cs360/labs/Lab-5-Tarc/bin/tarc**. However, with the exception of gradescripts 41-50, they will be legal tarfiles. With gradescripts 41-50, the tarfiles are bad, and your program must print an error message on standard error and exit. Your error messages do not have to match mine exactly.

## Ten Open Files

Same as the **tarc** lab.

## Useful System Calls

There is one more system call that you should be aware of:

- **utimes()**. Ok, so this is the world that we're in:
  - **st_mtime** is a **time_t**, and is in some operating systems' **struct stat** specifications, but not in others (although it's supported by a **#define**).
  - **st_mtimespec** is a **struct timespec**, which, if you look it up, has two fields to deal with sub-second times: **tv_sec** and **tv_nsec** (nanoseconds). Fortunately, that latter one fits into a 32-bit data type (although I'm not overly sure if that's what's used. I would be it's inconsistent).
  - **utimes()** has a second parameter which is a pointer to two **struct timeval**'s. Really? That one has two fields: **tv_sec** and **tv_usec** (microseconds).

I understand that this is a pain, but hopefully this prose helps you a little. You'll have to create an array of two **struct timeval**'s for **utimes()**. Go ahead and set the first one to the current time (I don't do any testing of the access time in this lab, so you don't have to worry about it). As for the second one, set the seconds to your stored **mtime**, and then set the microseconds to zero. Done.