Linux使用patchelf工具

1、下载patchelf工具

github 地址：https://github.com/NixOS/patchelf

下载地址：https://github.com/NixOS/patchelf/releases

我直接下载的是PatchELF 0.12

patchelf-0.12.tar.gz

2、安装patchelf工具

解压：

```
$ tar -axf patchelf-0.12.tar.gz
$ cd patchelf-0.12/
$ ls
bootstrap.sh  COPYING     Makefile.am       README.md    tests
BUGS          flake.lock  patchelf.1        release.nix  version
configure.ac  flake.nix   patchelf.spec.in  src
```

安装：

```
$ ./bootstrap.sh
./bootstrap.sh: 2: ./bootstrap.sh: autoreconf: not found
```

根据报错查找解决办法：

```
$ sudo apt-get install autoconf automake libtool
```

然后继续执行：

```
$ sh bootstrap.sh
autoreconf: Entering directory `.'
autoreconf: configure.ac: not using Gettext
autoreconf: running: aclocal --force --warnings=all
autoreconf: configure.ac: tracing
autoreconf: configure.ac: creating directory build-aux
autoreconf: configure.ac: not using Libtool
autoreconf: running: /usr/bin/autoconf --force --warnings=all
autoreconf: configure.ac: not using Autoheader
autoreconf: running: automake --add-missing --copy --force-missing --warnings=all
configure.ac:7: installing 'build-aux/compile'
configure.ac:5: installing 'build-aux/install-sh'
```

```
configure.ac:5: installing 'build-aux/missing'
                                                src/Makefile.am: installing 'build-aux/depcomp'
parallel-tests: installing 'build-aux/test-driver'
autoreconf: Leaving directory `.'
```

```
$ ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... no
checking for mawk... mawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for style of include used by make... GNU
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking dependency style of gcc... gcc3
checking for g++... g++
checking whether we are using the GNU C++ compiler... yes
checking whether g++ accepts -g... yes
checking dependency style of g++... gcc3
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating Makefile
config.status: creating src/Makefile
config.status: creating tests/Makefile
config.status: creating patchelf.spec
config.status: executing depfiles commands
```

然后是:

```
make
make install
```

3、验证安装

```
patchelf --version
```

4、使用

（1）查看依赖库

假设自己编译的动态库或可执行程序名字为demo，此处以demo为例。

```
$ readelf -d demo
Dynamic section at offset 0x7e3c30 contains 35 entries:
  Tag        Type                         Name/Value
 0x0000000000000001 (NEEDED)             Shared library: [libtorch.so]
 0x0000000000000001 (NEEDED)             Shared library: [libc10.so]
```

```
  0x0000000000000001 (NEEDED)             Shared library: [libIlmImf-2_2.so.22]

0x0000000000000001 (NEEDED)             Shared library: [libdl.so.2]
0x0000000000000001 (NEEDED)             Shared library: [librt.so.1]
0x0000000000000001 (NEEDED)             Shared library: [libtorch_cpu.so]
0x0000000000000001 (NEEDED)             Shared library: [libpthread.so.0]
0x0000000000000001 (NEEDED)             Shared library: [libstdc++.so.6]
0x0000000000000001 (NEEDED)             Shared library: [libm.so.6]
0x0000000000000001 (NEEDED)             Shared library: [libgcc_s.so.1]
0x0000000000000001 (NEEDED)             Shared library: [libc.so.6]
0x000000000000000f (RPATH)              Library rpath: [/home/wmz/Desktop/mypractice/libtorch/lib]
0x000000000000000c (INIT)               0x40c668  | 0x000000000000000d (FINI)               0xa1745c
  0x0000000000000019 (INIT_ARRAY)         0xdd3348
  0x000000000000001b (INIT_ARRAYSZ)       200 (bytes)
  0x000000000000001a (FINI_ARRAY)         0xdd3410
  0x000000000000001c (FINI_ARRAYSZ)       8 (bytes)
  0x000000006ffffef5 (GNU_HASH)           0x400298
  0x0000000000000005 (STRTAB)             0x4040b0
  0x0000000000000006 (SYMTAB)             0x400a08
  0x000000000000000a (STRSZ)              22774 (bytes)
  0x000000000000000b (SYMENT)             24 (bytes)
  0x0000000000000015 (DEBUG)              0x0
  0x0000000000000003 (PLTGOT)             0xde4000
  0x0000000000000002 (PLTRELSZ)           8232 (bytes)
  0x0000000000000014 (PLTREL)             RELA
  0x0000000000000017 (JMPREL)             0x40a640
  0x0000000000000007 (RELA)               0x409ff8
  0x0000000000000008 (RELASZ)             1608 (bytes)
  0x0000000000000009 (RELAENT)            24 (bytes)
  0x000000006ffffffe (VERNEED)            0x409e38
  0x000000006fffffff (VERNEEDNUM)         7
  0x000000006ffffff0 (VERSYM)             0x4099a6
  0x0000000000000000 (NULL)               0x0
```

（2）修改rpath

```
patchelf --set-rpath 'ORIGIN' demo
```

后来试了一下上面这样写不行，下面这样写才可以

```
patchelf --set-rpath '$ORIGIN/' demo
```

这样就添加了可执行程序demo所在路径为依赖库搜索路径。

参考：https://www.cnblogs.com/ar-cheng/p/13225342.html

官网使用说明：

README .md

PatchELF is a simple utility for modifying existing ELF executables and libraries. In particular, it can do the following:

- Change the dynamic loader ("ELF interpreter") of executables:

  ```
  $ patchelf --set-interpreter /lib/my-ld-linux.so.2 my-program
  ```

- Change the RPATH of executables and libraries:

```
$ patchelf --set-rpath /opt/my-libs/lib:/other-libs my-program
```

- Shrink the RPATH of executables and libraries:

```
$ patchelf --shrink-rpath my-program
```

This removes from the RPATH all directories that do not contain a library referenced by DT_NEEDED fields of the executable or library. For instance, if an executable references one library libfoo.so, has an RPATH /lib:/usr/lib:/foo/lib, and libfoo.so can only be found in /foo/lib, then the new RPATH will be /foo/lib.

In addition, the --allowed-rpath-prefixes option can be used for further rpath tuning. For instance, if an executable has an RPATH /tmp/build-foo/.libs:/foo/lib, it is probably desirable to keep the /foo/lib reference instead of the /tmp entry. To accomplish that, use:

```
$ patchelf --shrink-rpath --allowed-rpath-prefixes /usr/lib:/foo/lib my-program
```

- Remove declared dependencies on dynamic libraries (DT_NEEDED entries):

```
$ patchelf --remove-needed libfoo.so.1 my-program
```

This option can be given multiple times.

- Add a declared dependency on a dynamic library (DT_NEEDED):

```
$ patchelf --add-needed libfoo.so.1 my-program
```

This option can be give multiple times.

- Replace a declared dependency on a dynamic library with another one (DT_NEEDED):

```
$ patchelf --replace-needed liboriginal.so.1 libreplacement.so.1 my-program
```

This option can be give multiple times.

- Change SONAME of a dynamic library:

```
$ patchelf --set-soname libnewname.so.3.4.5 path/to/libmylibrary.so.1.2.3
```

patchelf支持的用法：

```
$ patchelf -h
syntax: patchelf
  [--set-interpreter FILENAME]
  [--page-size SIZE]
  [--print-interpreter]
  [--print-soname]      Prints 'DT_SONAME' entry of .dynamic section. Raises an error if DT_SONAME doesn't exist
  [--set-soname SONAME]     Sets 'DT_SONAME' entry to SONAME.
  [--set-rpath RPATH]
  [--remove-rpath]
  [--shrink-rpath]
  [--allowed-rpath-prefixes PREFIXES]       With '--shrink-rpath', reject rpath entries not starting with the allowed
  [--print-rpath]
  [--force-rpath]
```

```
  [--add-needed LIBRARY]    [--remove-needed LIBRARY]
  [--replace-needed LIBRARY NEW_LIBRARY]
  [--print-needed]
  [--no-default-lib]
  [--clear-symbol-version SYMBOL]
  [--output FILE]
  [--debug]
  [--version]
  FILENAME...
```

实际应用，设置运行库路径时，可以一次设置多个，中间用冒号隔开，如下，这条命令同时指定了三个路径，当前路径，上一级下的lib路径，以及cuda -10.2下的lib64路径：

```
$ patchelf --set-rpath 'ORIGIN:../lib/:/usr/local/cuda-10.2/lib64' dbnet_demo
```

经查看，这些依赖库都是可以正确找到的：

```
$ ldd dbnet_demo
    linux-vdso.so.1 (0x00007ffc96ce1000)
    libtorch.so => ../lib/libtorch.so (0x00007fb7014f9000)
    libc10.so => ../lib/libc10.so (0x00007fb701275000)
    libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007fb701071000)
    libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007fb700e52000)
    librt.so.1 => /lib/x86_64-linux-gnu/librt.so.1 (0x00007fb700c4a000)
    libtorch_cpu.so => ../lib/libtorch_cpu.so (0x00007fb6f0a89000)
    libtorch_cuda.so => ../lib/libtorch_cuda.so (0x00007fb6b5e89000)
    libnvToolsExt.so.1 => /usr/local/cuda-10.2/lib64/libnvToolsExt.so.1 (0x00007fb6b5c80000)
    libcudart.so.10.2 => /usr/local/cuda-10.2/lib64/libcudart.so.10.2 (0x00007fb6b5a02000)
    libstdc++.so.6 => /usr/lib/x86_64-linux-gnu/libstdc++.so.6 (0x00007fb6b5679000)
    libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007fb6b52db000)
    libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x00007fb6b50c3000)
    libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fb6b4cd2000)
    libgomp-7c85b1e2.so.1 => /home/wmz/Documents/pytorch-dbnet/build2/../lib/libgomp-7c85b1e2.so.1 (0x00007fb6b4aa800
    /lib64/ld-linux-x86-64.so.2 (0x00007fb702206000)
    libtensorpipe.so => /home/wmz/Documents/pytorch-dbnet/build2/../lib/libtensorpipe.so (0x00007fb6b45df000)
    libcudart-80664282.so.10.2 => /home/wmz/Documents/pytorch-dbnet/build2/../lib/libcudart-80664282.so.10.2 (0x00007
    libc10_cuda.so => /home/wmz/Documents/pytorch-dbnet/build2/../lib/libc10_cuda.so (0x00007fb6b412d000)
    libnvToolsExt-3965bdd0.so.1 => /home/wmz/Documents/pytorch-dbnet/build2/../lib/libnvToolsExt-3965bdd0.so.1 (0x000
```

运行可执行程序，也可以获得正确的结果：

```
$ ./dbnet_demo
ok
 file_names.size():5
0001004.jpg
./data/0001004.jpg
[W TensorIterator.cpp:918] Warning: Mixed memory format inputs detected while calling the operator. The operator will
The run time is: 0.408824s
CLOCKS_PER_SEC is: 1000000
0001001.jpg
./data/0001001.jpg
The run time is: 0.112061s
CLOCKS_PER_SEC is: 1000000
0001002.jpg
./data/0001002.jpg
```

```
The run time is: 0.100809s | CLOCKS_PER_SEC is: 1000000
0001003.jpg
./data/0001003.jpg
The run time is: 0.101038s
CLOCKS_PER_SEC is: 1000000
0000000.jpg
./data/0000000.jpg
The run time is: 0.095049s
CLOCKS_PER_SEC is: 1000000
```