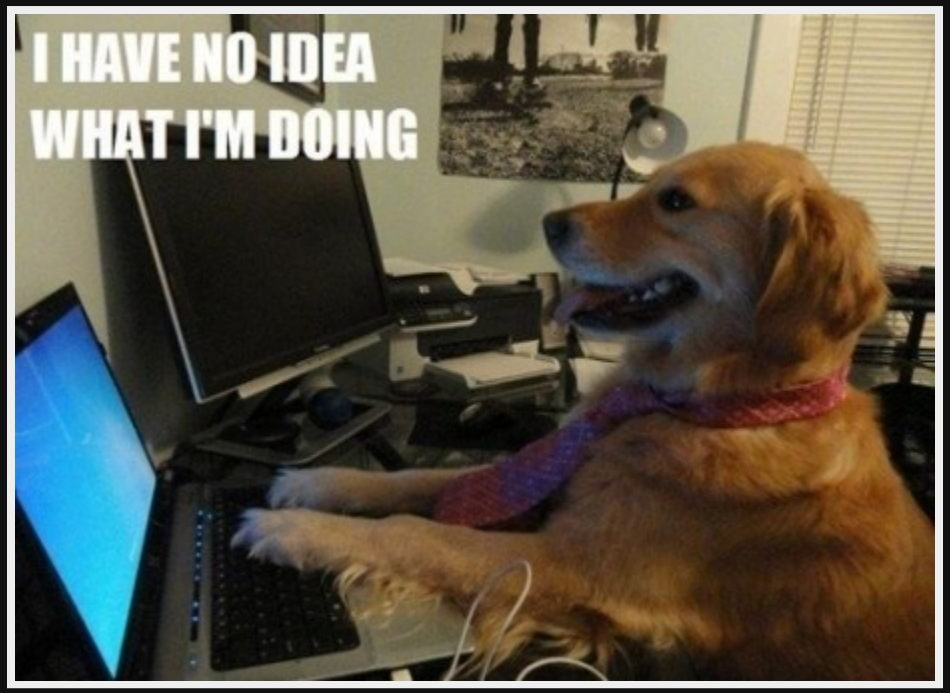
Shared Library Path Resolution in Linux

Danny Robson

Caveats



Mapping names to paths

```
danny@steve / $ ldd /bin/echo
    linux-vdso.so.1 (0x00007ffe37122000)
    libc.so.6 => /lib64/libc.so.6 (0x00007fc8e9fe4000)
    /lib64/ld-linux-x86-64.so.2 (0x00007fc8ea383000)
```

ld.so, ld-linux.so*

1. Find and load the shared objects needed by a program
2. Prepare the program to run
3. Run it

Don’t do this

'Just guess a path'

- /usr/lib
- /usr/lib64
- /usr/local/lib64
- /usr/lib64/OpenCL/vendors/intel
- /opt/nvidia-cg-toolkit/lib64
- /home/danny/prefix/usr/lib

The actual method

Look in directories from:

1. LD_PRELOAD environment variable
2. The DT_RPATH ELF section (if DT_RUNPATH doesn’t exist)
3. The environment variable LD_LIBRARY_PATH
4. The DT_RUNPATH ELF section
5. The cache file /etc/ld.so.cache
6. Probably /lib and /usr/lib (unless '-z nodeflib' linking was used)

See: `man ld.so`

/etc/ld.so.cache

Compiled from /etc/ld.so.conf, a priority list of

- directories
- include directives

Example

```
# ld.so.conf autogenerated by env-update; make all changes to
# contents of /etc/env.d directory
/usr/lib32/opengl/nvidia/lib
/usr/lib64/opengl/nvidia/lib
/lib64
/usr/lib64
/usr/local/lib64
/lib32
/usr/lib32
/usr/local/lib32
/lib
/usr/lib
/usr/local/lib
include ld.so.conf.d/*.conf
/usr/lib64/OpenCL/vendors/intel
/usr/lib32/qt4
/usr/lib64/qt4
/usr/lib/postgresql
/usr/lib64/postgresql
/usr/lib64/postgresql-9.5/lib64/
/opt/nvidia-cg-toolkit/lib32
```

rpath

Hard code a directory to search

$ORIGIN

Path to the binary

$LIB

'lib', or 'lib64'

$PLATFORM

Processor specific string (probably). eg, 'x86_64'

Implications

- If you assume library paths then your software *will* break (and some poor sod will have to spend 2 weeks fixing it…)
- Odd library locations are common;
  - nVidia OpenGL/OpenCL
  - Intel OpenCL
  - qt4, postgres, fltk
- Don’t emulate or guess. Use ldd, dlopen, etc to do the work for you.