Linux error while loading shared libraries: cannot open shared object file: No such file or directory

Asked 15 years, 11 months ago Modified 4 months ago Viewed 2.1m times



Program is part of the Xenomai test suite, cross-compiled from Linux PC into Linux+Xenomai ARM toolchain.

597





```
# echo $LD_LIBRARY_PATH
/lib
# ls /lib
ld-2.3.3.so
                   libdl-2.3.3.so
                                       libpthread-0.10.so
ld-linux.so.2
                   libdl.so.2
                                       libpthread.so.0
libc-2.3.3.so
                   libgcc_s.so
                                       libpthread_rt.so
libc.so.6
                   libgcc_s.so.1
                                       libstdc++.so.6
                                       libstdc++.so.6.0.9
libcrypt-2.3.3.so
                   libm-2.3.3.so
libcrypt.so.1
                   libm.so.6
# ./clocktest
./clocktest: error while loading shared libraries: libpthread_rt.so.1: cannot open
shared object file: No such file or directory
```

Is the .1 at the end part of the filename? What does that mean anyway?

linux shared-libraries file-not-found xenomai

Share Improve this question Follow

edited Jul 4, 2022 at 14:25

Neuron

5,793 • 5 • 43 • 62

asked Jan 26, 2009 at 18:07

zaratustra

8,698 • 9 • 38 • 42

- This might happen if you have recently installed a shared library and didn't run ldconfig(8) afterwards. Do 'ldconfig', there's no harm in it. AbiusX Jun 5, 2011 at 21:02
- +1 to @AbiusX comment running sudo Idconfig (assuming that libraries are in fact where they should be [/usr/bin/lib/, /usr/bin/include/, /usr/local/lib/ and /usr/local/include/ AFAIK], please correct me if I'm wrong) can resolve that problem. Cheers! AeroCross Nov 16, 2011 at 18:11
- @AbiusX I ran sudo Idconfig after compiling my program and it worked. Thanks! (The libraries were in /usr/local/lib.) kleinbottle4 Mar 7, 2021 at 10:54
- we need an update for this. its posted in 2009 for god sake its still happening greendino Jul 2, 2021 at 11:37
- what "update" do you need? there's three good answers to it, some of which may be applicable depending on your specific issue. zaratustra Feb 2, 2023 at 13:52

20 Answers

Sorted by: Highest score (default) \$



Your library is a dynamic library. You need to tell the operating system where it can locate it at runtime.

608

To do so, we will need to do those easy steps:



1. Find where the library is placed if you don't know it.



```
sudo find / -name the_name_of_the_file.so
```



2. Check for the existence of the dynamic library path environment variable (LD_LIBRARY_PATH)

```
echo $LD_LIBRARY_PATH
```

If there is nothing to be displayed, add a default path value (or not if you wish to)

```
LD_LIBRARY_PATH=/usr/local/lib
```

3. We add the desired path, export it and try the application.

Note that the path should be the directory where the path.so.something is. So if path.so.something is in /my_library/path.so.something , it should be:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/my_library/
```

Reference to source

Share Improve this answer Follow

edited Oct 20, 2023 at 20:09 gwr **10.8k** • 6 • 68 • 114 answered Jan 16, 2014 at 22:07



XOR 6.524 • 2 • 18 • 10

- The above mentioned answer was very clear, Thanks you first of all. I tried doing this in my Eclipse CDT Project Path (Lubuntu). /Debug\$ echo \$LD LIBRARY PATH /home/akhil/HDE/x86.linux/lib:/home/akhil/HDE/x86.linux/lib:. "/home/akhil/HDE/x86.linux/lib" this is where my libraries are actually available even, but still I get the same error. Any suggestions! - nahasapeemapetilon Jul 12, 2016 at 11:44 🧨
- 31 Try a "Idconfig" command after exporting your library. You might need to execute this command as "sudo". - XOR Sep 26, 2016 at 16:53
- 11 I believe LD_LIBRARY_PATH should point to the directory containing path.so.something, not to path.so.something itself. - gerrit Jun 23, 2017 at 15:47
- This is the right solution for the systems where sudo ldconfig cannot be executed, e.g. supercomputers. - Herpes Free Engineer Apr 16, 2018 at 19:22
- The export LD_LIBRARY_PATH command is essential so that other programs can read the environment variable. Without it only the shell can see the variable. - Zenul Abidin Feb 13, 2020 at 6:05



Here are a few solutions you can try:

265

Idconfig



As AbiusX pointed out: If you have just now installed the library, you may simply need to run ldconfig.



sudo ldconfig

Idconfig creates the necessary links and cache to the most recent shared libraries found in the directories specified on the command line, in the file /etc/ld.so.conf, and in the trusted directories (/lib and /usr/lib).

Usually your package manager will take care of this when you install a new library, but not always, and it won't hurt to run Idconfig even if that is not your issue.

Dev package or wrong version

If that doesn't work, I would also check out <u>Paul's suggestion</u> and look for a "-dev" version of the library. Many libraries are split into dev and non-dev packages. You can use this command to look for it:

```
apt-cache search <libraryname>
```

This can also help if you simply have the wrong version of the library installed. Some libraries are published in different versions simultaneously, for example, Python.

Library location

If you are sure that the right package is installed, and Idconfig didn't find it, it may just be in a nonstandard directory. By default, Idconfig looks in /lib, /usr/lib, and directories listed in /etc/ld.so.conf and \$LD_LIBRARY_PATH. If your library is somewhere else, you can either add the directory on its own line in /etc/ld.so.conf, append the library's path to \$LD_LIBRARY_PATH, or move the library into /usr/lib. Then run Idconfig.

To find out where the library is, try this:

```
sudo find / -iname *libraryname*.so*
```

(Replace libraryname with the name of your library)

If you go the <code>\$LD_LIBRARY_PATH</code> route, you'll want to put that into your <code>~/.bashrc</code> file so it will run every time you log in:

export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:/path/to/library

Share Improve this answer Follow

edited May 23, 2017 at 11:47



answered Feb 11, 2015 at 17:11



5 By default, /lib and /usr/lib but not /usr/local/lib? That has thrown me off several times over my career and wasted hours. – DarenW Mar 15, 2015 at 22:15

Adding .conf files of my own with the non-standard lib paths I need to /etc/ld.so.conf.d (pointed to by /etc/ld.so.conf) did the trick. — CivFan Jan 27, 2016 at 20:42

7 +1 for needing to run Idconfig. I wasn't using a package manager. I had to compile from source, so this was necessary. – Jeff Dec 9, 2016 at 17:06

Just like @CivFan, I added my own .conf in /etc/ld.so.conf.d . After which I (obviously) had to run ldconfig and it worked. I can feel you @DarenW. Its a pain in the butt – Hemil Sep 26, 2019 at 10:37

For anybody concerned: find -iname is the same as -name, but case insensitive. But be wary: it is not part of the standard, so could be absent in the implementation you use. — Cadoiz Oct 18, 2021 at 10:14



Update

153

While what I write below is true as a general answer about shared libraries, I think the most frequent cause of these sorts of message is because you've installed a package, but not installed the -dev version of that package.



Well, it's not lying - there is no libpthread_rt.so.1 in that listing. You probably need to reconfigure and re-build it so that it depends on the library you have, or install whatever provides libpthread_rt.so.1.

Generally, the numbers after the .so are version numbers, and you'll often find that they are symlinks to each other, so if you have version 1.1 of libfoo.so, you'll have a real file libfoo.so.1.0, and symlinks foo.so and foo.so.1 pointing to the libfoo.so.1.0. And if you install version 1.1 without removing the other one, you'll have a libfoo.so.1.1, and libfoo.so.1 and libfoo.so will now point to the new one, but any code that requires that exact version can use the libfoo.so.1.0 file. Code that just relies on the version 1 API, but doesn't care if it's 1.0 or 1.1 will specify libfoo.so.1. As orip pointed out in the comments, this is explained well here.

In your case, you *might* get away with symlinking <code>libpthread_rt.so.1</code> to <code>libpthread_rt.so</code>. No guarantees that it won't break your code and eat your TV dinners, though.

Share Improve this answer Follow

edited Sep 7, 2023 at 12:44

answered Jan 26, 2009 at 18:11





- 8 ... oh god, the .1 is part of the filename. Any idea what does it mean? zaratustra Jan 26, 2009 at 18:21
- 2 @zaratustra it means version Kossak May 6, 2022 at 11:34



You need to ensure that you specify the library path during linking when you compile your .c file:



gcc -I/usr/local/include xxx.c -o xxx -L/usr/local/lib -Wl,-R/usr/local/lib



The -w1, -R part tells the resulting binary to also look for the library in /usr/local/lib at runtime before trying to use the one in /usr/lib/.



Share Improve this answer Follow



answered Mar 23, 2016 at 16:29



- 8 This is the option I was looking for. Perhaps better would be -W1, -rpath DIR . jrw32982 Mar 31, 2017 at 20:14
- 1 This is the personal turning point in my programming-enthusiast life. It's awesome. Thank you! Max Herrmann Feb 16, 2023 at 16:31
- 1 This is excatly what I was looking for, thank you! Thiago Lages de Alencar Feb 7, 2024 at 18:15



Try adding LD_LIBRARY_PATH, which indicates search paths, to your ~/.bashrc file

35

LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:/path_to_your_library



It works!



Share Improve this answer Follow

edited Nov 14, 2019 at 18:53

Luce
325 • 3 • 13

answered Jul 7, 2014 at 13:21



Ankit Marothi **1,005** ● 11 ● 14

3 After that, the following line should be added to the .bashrc export LD_LIBRARY_PATH – user216652 Sep 2, 2020 at 23:28 /



The <u>linux.org reference page</u> explains the mechanics, but doesn't explain any of the motivation behind it :-(

14 For that, see <u>Sun Linker and Libraries Guide</u>



In addition, note that "external versioning" is largely obsolete on Linux, because symbol versioning (a GNU extension) allows you to have multiple incompatible versions of the same function to be present in a single library. This extension allowed glibc to have the same external version: [libc.so.6] for the last 10 years.

Share Improve this answer Follow

edited Oct 18, 2021 at 14:35

Cadoiz

1,656 • 1 • 24 • 33

answered Jan 27, 2009 at 6:01





Wanted to add, if your libraries are in a non standard path, run <code>ldconfig</code> followed by the path.

9

For instance I had to run:



sudo ldconfig /opt/intel/oneapi/mkl/2021.2.0/lib/intel64

to make R compile against Intel MKL



Share Improve this answer Follow

edited Oct 21, 2021 at 14:29

Cadoiz

1.656 • 1 • 24 • 33

answered Jun 21, 2021 at 15:51



sudo ldconfig <PACKAGE> worked for me. Thanks. – Maf Jul 18, 2023 at 6:21



cd /home/<user_name>/
sudo vi .bash_profile



add these lines at the end



LD_LIBRARY_PATH=/usr/local/lib:<any other paths you want> export LD_LIBRARY_PATH



Share Improve this answer Follow

edited Oct 29, 2017 at 18:54



answered Jun 21, 2016 at 17:36





Another possible solution depending on your situation.

If you know that libpthread_rt.so.1 is the same as libpthread_rt.so then you can create a symlink by:



ln -s /lib/libpthread_rt.so /lib/libpthread_rt.so.1



Then 1s -1 /lib should now show the symlink and what it points to.



Share Improve this answer Follow

edited Oct 6, 2015 at 5:00

answered Sep 15, 2015 at 23:58



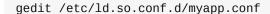
Could this be done for different version? In my case, I have libnsl.so.2, but my command is looking for libnsl.so.1. – Nathan Russell Feb 28, 2023 at 16:42

To add to this answer, you can see which directories the linker is looking for the libraries in with 1d --verbose | grep SEARCH_DIR and check for such near-matches. In my case, I had a problem where I had a libger-4.so.4.0.0 in /usr/lib but no libger-4.so.4, so all I had to do was create a symlink like this answer says and it fixed it. – Eric Pedley May 18, 2023 at 0:13



I had a similar error and it didn't fix with giving LD_LIBRARY_PATH in ~/.bashrc . What solved my issue is by adding .conf file and loading it. Go to terminal an be in su.







Add your library path in this file and save.(eg: /usr/local/lib). You must run the following command to activate path:



ldconfig

Verify Your New Library Path:

ldconfig -v | less

If this shows your library files, then you are good to go.

Share Improve this answer Follow

answered Jan 3, 2018 at 7:44





I had this error when running my application with Eclipse CDT on Linux x86. To fix this:



1. In Eclipse:



Run as -> Run configurations -> Environment



2. Set the path



LD_LIBRARY_PATH=/my_lib_directory_path

Share Improve this answer Follow



answered Oct 31, 2017 at 16:01





If you are running your application on Microsoft Windows, the path to dynamic libraries (.dll) need to be defined in the PATH environment variable.



If you are running your application on UNIX, the path to your dynamic libraries (.so) need to be defined in the LD_LIBRARY_PATH environment variable.



Share Improve this answer Follow

edited Oct 19, 2021 at 3:57

Bazer Con

105 • 4

answered Feb 27, 2016 at 12:33





I got this error and I think its the same reason of yours



error while loading shared libraries: libnw.so: cannot open shared object file: No such file or directory



Try this. Fix permissions on files:



cd /opt/Popcorn (or wherever it is)
chmod -R 555 * (755 if not ok)

Share Improve this answer Follow

edited Jun 19, 2018 at 18:02

answered Jun 19, 2018 at 17:47

Sa Salmi Ahmed



Ahme **21** • 3



Try to install *lib32z1*:

2 sudo apt-get install lib32z1 Share Improve this answer Follow edited Sep 23, 2021 at 23:07 answered Jan 7, 2014 at 14:34 zajac.m2 Victor **8.862** • 5 • 17 • 36 **1.238** • 14 • 15 I use Ubuntu 18.04 Installing the corresponding -dev package worked for me, 2 sudo apt install libgconf2-dev Before installing the above package, I was getting the below error: A) turtl: error while loading shared libraries: libgconf-2.so.4: cannot open shared object file: No such file or directory Share Improve this answer Follow edited Oct 20, 2021 at 23:22 answered Jun 2, 2020 at 17:06 Bazer Con prabhugs **105** • 4 **742** • 8 • 20 All I had to do was run: sudo apt-get install libfontconfig1 1 I was in the folder located at /usr/lib/x86_64-linux-gnu and it worked perfectly. Share Improve this answer Follow answered Mar 16, 2015 at 13:56 jonny **3,088** • 1 • 20 • 34 The error occurs as the system cannot refer to the library file mentioned. Take the following steps:



1. Running locate libpthread_rt.so.1 will list the path of all the files with that name. Let's suppose a path is /home/user/loc.



2. Copy the path and run cd home/USERNAME. Replace USERNAME with the name of the current active user with which you want to run the file.

- 3. Run vi .bash_profile and at the end of the LD_LIBRARY_PATH parameter, just before . , add the line /lib://home/usr/loc:. . Save the file.
- 4. Close terminal and restart the application. It should run.

Share Improve this answer Follow







I got this error and I think its the same reason of yours

0

error while loading shared libraries: libnw.so: cannot open shared object file: No such file or directory



Try this. **Fix permissions** on files:



sudo su
cd /opt/Popcorn (or wherever it is)
chmod -R 555 * (755 if not ok)
chown -R root:root *

Share Improve this answer Follow

edited Oct 18, 2021 at 11:22

answered Apr 7, 2018 at 9:36





A similar problem can be found here. I've tried the mentioned solution and it actually works.

0

The solutions in the previous questions may work. But the following is an easy way to fix it. It works by reinstalling the package libwbclient in fedora:



dnf reinstall libwbclient



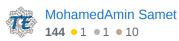
Share Improve this answer Follow

edited Oct 19, 2021 at 11:41

Denis

2,314 • 3 • 24 • 24

answered Jul 10, 2019 at 0:55





Reason/Solution

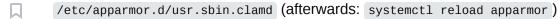
Apparmor:

0

In newer Debian version (I think >= Debian11) apparmor is configured when installing clamav from the debian repo.



If clamav is not completely uninstalled before reinstalling from e.g. <u>clamav deb package</u>, or anything is changed in the configuration (e.g. clamav definition path) it has to be also adjusted in





A weak but instant workaround would be to add flags to relax apparmor, so it logs and complains about access to non-regular places, but won't fail hard:

```
/usr/local/sbin/clamd flags=(complain,attach_disconnected){
```

tldr

I had this problem in 2024 with clamav:

/usr/local/sbin/clamd: error while loading shared libraries: libclamav.so.11: cannot open shared object file: No such file or directory

Although the permissions in /usr/local/lib where correct, strace /usr/local/sbin/clamd showed:

openat(AT FDCWD, "/usr/local/lib/libclamav.so.11", O_RDONLY|O_CLOEXEC) = -1 **EACCES** (Permission denied)

Share Improve this answer Follow

answered Sep 10, 2024 at 7:25





Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.