共享的动态链接库

对于本练习,您将尝试在 Linux 上使用共享的动态链接库。你会自己写
一点点共享库,看看如何配置系统以使用动态链接的最新兼容版本
库而无需重新编译客户端代码。

您需要在其中一台 EOS Linux 机器上进行此练习。我已经编译了一些组件
您将需要这些系统。它们可能适用于其他 Linux 机器（它们适用于我）,但您可能会在不同系统上遇到兼容性问题。另外,您将为此提交编译后的二进制文件
锻炼。在 EOS Linux 机器上构建它有助于确保您的解决方案在我试用时适用于我。

登录 EOS Linux 机器后,创建一个目录,您计划在该目录中处理此项目。改成
该目录并使用以下命令复制您需要的文件：

cp /mnt/coe/workspace/csc/CSC246/001/ex09/* 。

您刚刚复制了一个名为 libhello.so.1.0 的共享库,以及一个声明了
库中定义的一个函数。该函数称为 getMessage()。它返回一个动态分配的字符串
包含一条 hello world 消息。这个例子很简单,但它是共享库可能如何分布的典型例子。
头文件描述了库中可用的内容,因此可以编译客户端代码以使用该库。图书馆
文件 (libhello.so.1.0) 包含编译后的代码。它由使用该库的客户端代码在启动时自动加载。您没有获得库的源代码;您实际上不需要
它来使用该库。

您还可以获得名为 driver.c 的测试驱动程序的源代码。它调用由
库,打印出它收到的消息,为消息释放内存,然后退出。

准备和使用图书馆

在 Linux 机器上,共享库通常由管理员安装在标准位置。我们不去
在 EOS Linux 机器上执行此操作（我们不能）,因此我们需要配置我们的环境,以便它寻找共享
当前目录中的库。使用以下命令设置 LD_LIBRARY_PATH 环境变量
应该为你做这件事。你只需要这样做一次,但如果你退出并重新登录以稍后完成练习,
你需要再做一次：

$ 导出 LD_LIBRARY_PATH=。

如果您查看库的名称,您会在末尾看到一些版本控制信息。这用于允许系统

找到库的最新兼容版本。只要 .so 之后的数字（本库的 .1)是

同样，图书馆被认为是兼容的。下一个值（此库的 .0)称为次要编号。它用于将库的较新、兼容版本与旧版本区分开来。

通常，共享库将被配置为显示在几个不同的文件名下，不太具体

版本控制信息。这是通过符号链接完成的。这些就像文件系统中的指针。当您尝试打开一个符号链接时，操作系统会注意到它是一个符号链接并打开它指向的

文件。跑步

以下命令为库创建两个符号链接。第一个链接说文件 libhello.so.1.0 是

库版本 1 的最新版本。所以，如果你想加载 libhello.so.1，那么文件 libhello.so.1.0 就是那个

你应该使用。下一个链接在编译时使用。它说，如果你正在编译一个使用这个库的程序，并且

你不关心你使用的是什么版本，那么你将获得 libhello.so.1.0 版本。

```
$ ln -sf libhello.so.1.0 libhello.so.1
$ ln -sf libhello.so.1.0 libhello.so
```

现在，使用 ls 看看你做了什么。您应该会看到如下内容。我们刚刚创建的两个新文件

实际上是符号链接，指向包含当前版本库的文件：

```
$ ls -l
共计 11
-rwxrwx---。 1 sjiao2 NCSU 278 Mar 1 16:20 driver.c -rwxrwx---。 1 sjiao2 NCSU 121
Mar 1 16:20 hello.h lrwxrwx---。 1 sjiao2 NCSU 15 Mar 1 16:24 libhello.so ->
libhello.so.1.0 lrwxrwx---。 1 sjiao2 NCSU 15 Mar 1 16:24 libhello.so.1 -> libhello.so.1.0 -rwxrwx---。 1 sjiao2 NCSU 6198 3
月 1 日 16:20 libhello.so.1.0
```

现在，我们应该能够编译并运行我们的驱动程序了。下面的命令编译它。你应该

已经识别出这些 gcc 选项中的大部分。 -L。最后说要在当前目录中查找库，并且

-lhello（小写的 L)表示链接到名为 libhello.so 的库。这就是其中一个符号链接

我们在上面创建的被使用了。如果删除名为 libhello.so 的链接，此编译将失败。

```
$ gcc -Wall -std=c99 driver.c -o 驱动程序 -L。 -你好
```

现在，在运行这个程序之前，让我们使用 ldd 查看它所依赖的共享库。在

可执行文件应该给你一个像下面这样的报告。你可以看到我们的程序会尝试加载 libhello.so 的版本 1

当它开始时。这就是我们创建的另一个符号链接将要使用的地方。如果我们删除了 libhello.so.1 链接，我们的程序将无法在启动时找到它需要的库。

```
$ ldd ./驱动程序
```

```
linux-vdso.so.1 => (0x00007ffd772df000) libhello.so.1 => ./
libhello.so.1 (0x00007f84b2d76000) libc.so.6 => /lib64/libc.so.6 (0x00007f84b29a8000) /
lib64/ ld-linux-x86-64.so.2 (0x00007f84b2f77000)
```

现在,最后,让我们尝试运行该程序。你应该得到标准的“Hello World”。信息。那是字符串

getMessage() 返回我给你的库的版本:

```
$./司机
你好世界。
```

更新库

现在,让我们用更新的版本替换库。我们应该能够让驱动程序使用更新版本的库,而无需重新编译驱动程序。这才是重点;无需重新编译,客户端程序应该

能够自动使用更新的、动态链接的库,只要它们与它们的版本兼容

被编译。

编写您自己的库版本。创建一个名为 hello.c 的文件,包含 hello.h 头文件并定义你自己的

getMessage() 函数。您的函数将需要为您要返回的消息分配 malloc() 空间,填写

带有空终止字符串的消息,然后返回它。对于您的库版本,返回字符串

“Hello unity-id.”,使用你自己的统一 ID。因此,例如,如果我正在做这个练习,我的函数将返回

“你好sjiao2。”

要编译您的库版本,您需要在 gcc 行中提供 -fPIC 选项。这告诉编译器编写与位置无关的代码,因此库的代码不必占用相同的逻辑地址范围

对于使用该库的每个程序。

```
$ gcc -Wall -fPIC -c hello.c
```

上面的命令为您的库代码创建了一个目标文件。您需要获取链接器才能创建共享库

从这个对象。通常,这是您将大量库文件合并到一个库中的地方,但是共享库

我们正在做的是微不足道的,只包含一个只有一个功能的对象。在下面的命令中,我们告诉 gcc

构建共享库并将其写入名为 libhello.so.1.1 的输出文件(因此,同一库的更新版本)。

-Wl 选项和紧随其后的语法(小写的 L)告诉链接器我们正在构建一个

应与版本 libhello.so.1 兼容的库。

```
gcc -shared -Wl,-soname,libhello.so.1 -o libhello.so.1.1 hello.o
```

为了让我们的驱动程序使用更新版本的库,我们需要更新符号链接,以便它们指向

库的新版本。真的,您只需要替换这些链接中的第一个,但我们不妨同时保留它们

持续的:

```
$ ln -sf libhello.so.1.1 libhello.so.1
$ ln -sf libhello.so.1.1 libhello.so
```

在此之后,您应该看到这两个符号链接现在指向库的最新版本,即您

上面创建的。检查新文件列表应该会显示如下内容:

```
$ ls -l
总计 31
-rwxrwx--x。 1 sjiao2 NCSU 18176 Mar 1 16:27 驱动程序 -rwxrwx---。 1 sjiao2 NCSU 278 Mar
1 16:20 driver.c -rwxrwx---。 1 sjiao2 NCSU 187 Mar 1 16:30 hello.c -rwxrwx---。 1 sjiao2 NCSU
121 Mar 1 16:20 hello.h -rwxrwx---。 1 sjiao2 NCSU 1712 Mar 1 16:30 hello.o lrwxrwx---。 1
sjiao2 NCSU 15 Mar 1 16:32 libhello.so -> libhello.so.1.1 lrwxrwx---。 1 sjiao2 NCSU -rwxrwx---。
1 sjiao2 NCSU 6198 3 月 1 日 16:20 libhello.so.1.0 -rwxrwx--x。 1 sjiao2 NCSU 8320 Mar 1 16:31
libhello.so.1.1
1 月 15 日 16:32 libhello.so.1 -> libhello.so.1.1
```

看。这些指向您刚刚创建的新库版本。

现在,尝试再次运行驱动程序。你还没有重新编译它(希望如此),但它应该自动获得

启动时更新的库。这是我得到的:

```
$./司机
你好 sjiao2。
```

对于本练习,只需将您的共享库 libhello.so.1.1 提交到名为 EX09 的 Gradescope 作业。通常,

您被要求提交源文件而不是二进制文件,但是,在这种情况下,提交库将有助于证明

你做了练习。我将通过链接不同的驱动程序并查看它是否有效来对其进行评分。我也会找

对于您的 unity ID,请确保您正确设置了这一部分。