Python利用pyd防止反编译教程

由于py文件打包生成的可执行文件很容易被反编译，而利用Cython将py文件转成的pyd文件只能被反汇编，不能被反编译，因此利用pyd文件将核心代码封装起来，可以保护软件不被滥用。

### py文件转pyd文件

py文件转pyd文件要注意以下问题：

1、一个py文件只能转成一个pyd文件。

2、最好只把关键代码转成pyd，如果把全部文件都转成pyd，只留一个入口py文件，后续打包需要手动添加很多依赖，非常麻烦，甚至可能失败。

3、py文件中不能有字节对象，即不能用一个字节对象来给一个变量名赋值，如果需要使用字节对象，可以将字节对象的值作为函数值进行传递（例如函数a（）的返回值是一个字节对象，不要创建一个变量b = a（），如果c函数要接收一个字节对象，直接使用c（a（））就行了）。

4、第四点是最关键的，打包pyd文件的思路决定了你的软件是否被成功加密了。因为pyd文件只能保护里面的源码不被破解，但是pyd文件是可以被正常调用的。由于不能将项目所有文件全部转成pyd，因此破解软件的人是一定可以调用你的pyd文件的。所以我们应该在核心pyd文件中加判断用户状态的操作，而调用的判断用户状态的函数应该包装在另一pyd文件中，这样即使pyd文件被调用，软件还是能根据用户状态来决定是否要运行核心代码。

以下是py转pyd的代码：

import os  
import shutil  
import time  
from distutils.core import setup  
from Cython.Build import cythonize  
  
# 在file\_path填写需要转换的文件名  
file\_path = "auth.py"  
setup(  
 name='any words.....',  
 ext\_modules=cythonize(file\_path, compiler\_directives={'language\_level': 3})  
)  
time.sleep(2)  
# 删除生成的c语言文件  
filename = file\_path.split('.py')[0]  
os.remove('%s.c' % filename)  
# 删除生成的build文件夹  
build\_folder\_path = os.path.join(os.getcwd(), 'build')  
shutil.rmtree(build\_folder\_path)

将该代码文件命名为build\_pyd.py,放入需要转成pyd文件的同级目录，在终端切换到该目录，输入：python build\_pyd.py build\_ext –inplace 回车即可生成pyd文件。Python调用pyd和py文件的规则是一样的，都是根据文件名进行调用，并且pyd与py文件名相同时，优先调用pyd文件。

### 利用pyinstaller打包

在把相关的py文件转成pyd之后，可以运行软件，如果可以成功运行就可以进行打包工作了。

pyinstaller打包不会去检测pyd模块内部引用了哪些库，所以这就需要我们手动添加pyd文件内部引用的依赖。手动添加依赖的方法：打开cmd窗口，切换到软件目录，输入软件可执行文件名然后回车，这时候cmd窗口会打印报错信息，告诉你哪个文件导入失败，这时候就去conda环境的文件夹中（一般在Lib文件夹或者Lib的site-package文件夹中）去找就行了，找到了就直接复制到软件目录中。

如果运气好，手动依赖确实能解决问题，而有些文件我们可以找到并复制到软件目录，这些文件它又引用了一大堆dll文件，而每个dll文件可能又引用了很多别的dll文件，总之这时候再去手动复制工作量就太大了，甚至完成不了。

这里我发现了一个比较好的方法：

1、先将原工程（所以py文件都未转成pyd文件）用pyinstaller打包，这时候pyinstaller会将项目所有用到的py文件都封装到可执行文件exe中，而项目中引用到的非py文件（如dll文件等），它都会打包到源工程的软件目录中，这样直接打包后基本上都是能直接运行的。

2、我们再将新工程（核心代码已经转成pyd文件）用pyinstaller打包，打包后生成新工程的软件目录。这时候把源工程软件目录的所有文件（除了exe可执行文件）全部复制到新工程的软件目录（这样做的原理就是这两个工程引用的非py文件依赖一定是一样的，不同的是源工程中引用的py文件都在exe中了，而新工程引用的某些py文件由于被pyd屏蔽了，暂时还未加入到新工程软件目录中。）。这样就解决了非py文件依赖的问题，在这种情况下再去用cmd窗口执行，还会遇到报错，但这些报错都是告诉我们缺少了哪些py文件（或者py文件夹），在conda环境文件夹中或者项目文件夹中找到，复制过来就行了。