教你如何截获Oracle数据库连接密码 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/143/2021_2022__E6_95_99_E4_BD_A0_E5_A6_82_E4_c102_143425.htm 大部分的服务器/客户端系统的结构可以这样描述： 客户端 对于这些系统，一般的安全问题出在由(2)所示的地方，比如说当使用 POP3 协议收取邮件，或者用 Telnet 登录到远程主机的时候，其登录密码都是未经加密的，只要在网络上安装一个嗅探器 (Sniffer) 来监听数据包，就可以很容易地截获用户名和密码。 但对于 Oracle 系统来说，用户名和密码在网络上传递之前，是经过加密的，而且加密的算法是不可逆的，即使使用嗅探器探听到数据包，开始无法把数据库的连接密码恢复出来，Oracle 系统的结构可以如下描述： 客户端应用程序 对于这一类系统，所有在(2)或者(3)处监听到的登录数据包都是已经经过加密的，但是，考虑一下我们编写 Oracle 数据库应用程序的时候，无论是通过 ODBC 还是 Pro C，或者其他的 BDE 环境等，都是将数据库连接的用户名和密码用明文的方式传递给 Oracle 客户端驱动程序的，所以在(1)位置的数据流肯定明文的，密码是在 Oracle 客户端软件中被加密后才经过(2)、(3)等步骤发送出去，如果在(1)的位置进行拦截，就可能拦截到密码。 考虑到步骤(1)发生在应用程序到 Oracle 系统的调用中，也就是发生在 API 调用的层次，所以只要找到密码加密模块的入口，在对相应的 API 进行 Hook，就能截获到密码了。 有人可能存在一个疑问：使用 Sniffer 可以监听到网络上其他计算机的连接数据包，而在 API 层次上进行拦截是针对本机的，但要是自己能够在本机上连接，就表示已经知道密码了

，再去截获不是多此一举吗？ 非也！ 实际上大部分的 Oracle 应用程序都包括一个用户开发的客户端，这个客户端可能是用 C、PowerBuilder 和其他语言开发的，这些软件提供一个界面提示用户输入用户名和密码登录系统，但是这个用户名和密码并不是数据库的连接用户名和密码，而仅仅是一个类似于 users 表中的一条记录而已，而程序内部内置的数据库连接帐号才是我们的目标，一般来说，客户端应用程序是这样工作的： 1. 使用一个内置的数据库连接帐号连接到数据库。 2. 弹出一个对话框提示用户输入用户名 xxx 和密码 yyy 3. 使用类似于 0select * from users where username=xxx and password=yyy 一类的 SQL 语句查询用户是否有权登录系统。 我们的目标就是步骤1中的连接帐号，这个帐号存在于客户端软件中，虽然可能已经被静态加密（也就是说用16进制软件去搜寻可执行文件时并不能被找到），但它运行后需要连接数据库的时候必然会被解密并用明文传递到 Oracle 客户端软件中。 方法 好了，现在来看看具体的实现方法。 1. 相关的调用 第一步当然要知道在哪里下手，经过了一番跟踪以后（这里省去跟踪的步骤 n 步，大家可以尝试自己跟踪一下），就可以发现用户名和密码是在 OraCore8.dll 模块中的 lncupw 函数中被加密的，而且这个函数的调用方法如下： invoke lncupw,addr Output,1eh,addr szPassword,dwLenPass,addr szUserName,dwLenName,NULL,1函数的入口参数包括明文的数据库连接用户名和密码，以及他们的长度，运行的结果是在第一个参数Output指定的缓冲区中返回加密后的数据，以后这个加密后的数据会被发送到服务器端进行认证。 2. 具体的实现方案 我们的方法就是在对 OraCore8.dll 进行补丁，在

dll 文件中附加一段代码，然后修改 dll 的导出表中 lncupw 函数对应的入口地址，将它指向到附加的代码中，然后由这段代码在堆栈中取出用户名和密码并显示出来，完成这个步骤后再跳转到原始的 lncupw 函数的入口地址去执行原有的功能。 这个方案涉及到两个技术问题，第一是对 dll 文件的修改问题，这个问题可以归结为在 PE 文件后添加可执行代码的方法问题，第二就是写被附加到 dll 文件后的程序体的问题。 对 dll 文件的修改代码的片断如下，在这以前，我们假定已经做了其他这样一些工作： ※ 文件名字符串放在 szFileName 指定的缓冲区中。 ※ 已经对文件进行校验，找到了导出表中的 lncupw 项目，这个项目在文件中的 Offset 放在 dwOffsetPeHeand 中，lncupw 的原始入口RVA放在 dwProcEntry 变量中。 