oracle认证辅导:重访Oracle密码Oracle认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/584/2021_2022_oracle_E8_AE_A4_E8_c102_584567.htm 对于那些对Oracle关系数据库系统的安全问题，特别是涉及到Oracle密码机制或算法的问题非常关注的人来说，老版本Oracle(特别是10g及其以下的版本)一直被认作是黑客们容易得手的攻击目标。似乎你永远不可能找到能百分之百保护你的系统免受黑客攻击的办法。你可以从很多方面来武装你的系统，但是有一些事情你是永远不可能避免的，例如总会有人有需要必须能够接触到并存取敏感数据，绝大多数的客户端连接也都会涉及到通过网络来传输数据。 有时候数据库管理员也不得不动用一下“ 黑客 ”的密码破解技巧才能解决问题。你会问，数据库管理员本来就有数据库内部所有的密钥，又怎么摇身一变成了黑客呢?因为能够存取所有的数据并不意味着可以查看所有的数据。特别是，能够查看密码的哈希值(hashed value)，并不代表就能看到密码本身。 那么数据库管理员为什么会想要看到实际的密码值呢?更确切的讲，为什么数据库管理员想要知道某个特定的密码明文呢?你可以想象很多情况下确实是有这个需要，常见的原因包括老应用产品的使用，高频率的人员调动，原本的密码管理和密码归档没有做好等等。改变SYS和SYSTEM密码通常不是什么大问题，但是如果是OLD_APP模式的密码呢?在网上搜索“ Oracle密码破解工具 ”，你会找到不少“ 好东西 ”，甚至还能找到自制的类似于黑客程序的软件。本文选择了Laszlo Toth的woraauthbf工具，从以下链接(http://www.soonerorlater.hu /index.khtml?article_id=513)可以

浏览并下载，该程序能很好的满足本文的需要。你可以先用woraauthbf创建一个包括用户名、哈希密码值、SID和服务器名的文本文件来对付Oracle的老版本看看。提供的这几项中只要用户名和哈希密码值是真的就行了。你如果深入研究过Oracle是怎么创建哈希值的，你肯定会清楚用户名和密码是紧密连锁的，而SID和服务器名和哈希值的创建没有任何关系。其他的一些“ 破解 ”程序依赖于网络信息，例如客户端、服务器IP地址、端口和第三方“ 嗅探器 ”工具来窥视在客户端和服务器之间传送的数据。 还是快速进入实例吧。复制下面指令的输出结果到一个txt文件，这样就创建了上文所说的密码文件。 0select username||:||password||:||name||:||host_name||: from sys.dba_users, sys.V_$DATABASE, sys.v_$instance. 再次提醒一下，上面的name和host_name随便你怎么取，或者用真实值也行。本例子的输出文本文件内容如下： SCOTT:DE59105EDBF4A687:ORCL:MYPC: 我们知道Oracle测试用户Scott的密码为tiger，这里为tigers(最后输出的结果实为TIGERS，Oracle忽略大小写)，从5字符变为6字符。解压缩上面下载的woraauthbf文件后，打开命令提示行(DOS)窗口，从这里调用该工具。将密码文件名存为“ named password_file.txt ”， 命令行文本输入如下： woraauthbf.exe -p c:\password_file.txt 所有参数选用默认，执行完本次会话输出结果如下所示： C:\[my path]&gt.woraauthbf.exe -p c:\password_file.txt Usernames will be permuted! The number of processors: 2 Number of pwds to check: 321272406 Number of pwds to check by thread: 160636203 Password file: c:\password_file.txt, charset: alpha, maximum length: 6, type: hash

Start: 0 End: 160636203 Start array thread with 489 number of passwords! Start: 160636203 End: 321272406 Writing session files... Writing session files... Password found: SCOTT:TIGERS:ORCL:MYPC Elpased time: 164s Checked passwords: 153976754 Password / Second: 938882 该程序计算了3亿2千1百多万需要检查的密码，而且使用了两个处理器把工作量减半了，而且默认字符集为alpha(A-Z)，也需要花费了164秒才能确定Scott的密码为TIGERS，每秒钟检查的密码个数为938,882。我们在检查了差不多一半的密码就很幸运的中标了。 如果排除物理因素的限制(CPU数量和处理器速度等)，那么影响运行完成时间的关键因素有两个：密码长度和字符集。如果你知道密码长度和字符集 (纯字母还是字母加数字还是字母数字加特殊字符)，你就能够大大减少需要检查的密码数量。一开始就缩小猜测范围当然就能够显著减少运行时间。为了做个对比，我们把字符集改为alphanum，Scott的密码不变，需要花费6分多钟才能找到Scott的密码。如果在前面所用的同一个用户密码文件中添加另外一个用户的密码信息，假设也知道密码长度同为6字符，且是字母加数字类型的，那么整个运行时间超过了29分钟(由于隐私的原因，下面所示的第二个用户的名字和密码都已经编辑过了)。 woraauthbf.exe -p c:\password_file.txt -m 6 -c alphanum Usernames will be permuted! The number of processors: 2 Number of pwds to check: 2238976116 Number of pwds to check by thread: 1119488058 Password file: c:\password_file.txt, charset: alphanum, maximum length: 6, type: hash Start: 0 End: 1119488058 Start: 1119488058 End: 2238976116 Start array thread with 490 number of passwords!

Writing session files... Writing session files... Writing session files... Writing session files... Writing session files... Writing session files... Password found: SCOTT:TIGERS:ORCL:MYPC Writing session files... Writing session files... Writing session files... ... Writing session files... Writing session files... Password found: SOMENAMES:X1M72Y:ORCL:MYPC Elpased time: 2152s Checked passwords: 1917149967 Password / Second: 890868 上述密码文件中的第二个条目来自一个8i数据库系统，而Scott的哈希值来自10g版本。本文的寓意非常明确：保护好任何会显露用户名及其哈希密码值的东西，不要轻易让其被他人存取，特别是SYS.USER$表(不要依赖DBA_USERS视图)。 查找第二个用户的密码明文所花费的运行时间很合理。 有时候你可能要花费好几个小时(甚至好几天)来查找某个密码，不过这比起和那些讨厌停机的用户一起修改已经忘掉的密码花费的成本要低多了。