数据库安全性：超越口令计算机等级考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/619/2021_2022__E6_95_B0_E6_8D_AE_E5_BA_93_E5_c98_619331.htm 编辑特别推荐: 全国计算机等级考试（等考）指定教材 全国计算机等级考试学习视频 全国计算机等级考试网上辅导招生 全国计算机等级考试时间及科目预告 百考试题教育全国计算机等级考试在线测试平台 全国计算机等级考试资料下载 全国计算机等级考试论坛 近来，公众和企业对于保护私有和私人信息的意识有了显著的增强。随着许多国家/地区现在出台了特定的法规，保护个人信息数据现在已经不仅仅是项公共关系事务，而且还是一项法定义务。 无论如何，保护 IT 系统(无论是在事务处理 (OLTP) 还是在数据仓库环境中)中的机密数据都是企业运营的首要考虑事项。例如，您能想象到一个在数据库中未存储客户姓名、地址和信用卡卡号的销售系统吗?私人数据是当前系统的战略资产，因此公司应采取一种具有积极和健壮的综合方法，通过安全策略实施来保护机密数据。从这个角度看来，组织的战略和战术决策必须以最终结果为导向，而不是集中于特定项目或当前的业务需要，从而避免重新设计而带来的高成本甚至是失去客户。 通常会采用许多复杂措施阻止在网络和操作系统级的未授权访问，并且将其集成到非定制或定制的应用程序系统中。而实际存储信息的的数据库却往往只使用标准的用户名/口令机制进行保护。Oracle 数据库 10g 对这一机制的实现是最好的，即使这样，如果口令泄密，那么保护也将随之不再存在了。Oracle 数据库可以通过 Oracle 虚拟专用数据库、Oracle 标签安全和其它机制提供更多保护

，但这些机制在实际生产中仍未得到充分应用。 在这篇技术文章中，我将介绍(并通过演示说明)在假定一个或多个数据库口令已经泄露的情况下，如何实现安全机制。此方法提供了一种简单的方法来组合使用 Oracle 数据库 10g 第 1 版中安全特性(Oracle9i 中包含有其中的一部分特性)，使得在入侵者即使建立了数据库连接的情况下，仍能实现高级别的保护。其主要目的是避免机密数据遭到未授权用户(无论该用户是外来的黑客还是公司内部的数据库管理员)的破坏。所提供的示例专用于事务环境，但这于原理同样可应用于商务智能和数据仓库环境中。 数据库安全性目标 Oracle 数据库是实际安全实施的重要组成。一般说来，运行 Oracle 数据库引擎的服务器得到了防火墙的很好保护，但这并不能排除未授权访问尝试(包括内部员工的访问尝试)的可能性。除了传统的用户名/口令方法之外，Oracle 数据库引擎还提供了自身的安全机制，以便即使在通过了其他所有安全障碍的情况下，仍能保护它的数据内容。以下各部分中确定的安全措施都假定入侵已渗透到数据库级别，这些措施将用作数据库自身的最后一道防线，但它们并不能用来代替外部保护。 在假定所有其他安全措施已被绕过，并且未授权数据库访问已经开始了的前提下，以下各部分定义的解决方案都用于构建数据库防御特性，以确保： Oracle 应用服务器(作为数据库的安全客户端)可以在需要时读取、插入和更新所有数据。Oracle 应用服务器将使用它的内部安全机制和应用程序专用的安全机制来确保私人数据免遭表示层中的未授权用户的入侵。 使用 SQL*Plus，在错误解决过程中能够进行安全的数据库访问，包括能够查看机密信息。 其他数据库访问无法检索私人客户端信息。

演示安装 本练习包含一个典型的销售类型数据模型，其中要保护的数据存储在 CUSTOMER 数据库中，具体而言是 CARD_NO 列中。该示例使整个表对未授权请求显示为空，因此进行 SELECT * from CUSTOMER. 将检索不到任何记录。一个表面上看起来不包含任何记录的表将比包含记录但将“令人感兴趣”的列隐藏或屏蔽起来的表更不会引起入侵者的关注(他们认为前者可能根本未被使用)。 但对 DBMS_RLS.ADD_POLICY 调用稍微进行修改后，此解决方案将隐藏(显示为 NULL)或屏蔽(显示为 ****)受保护列 CARD_NO 的值，但显示其他列的值的记录。可以通过在 DBMS_RLS.ADD_POLICY 调用中指定 sec_relevant_cols 和 sec_relevant_cols_opt 参数来实现功能。本文的支持文件中的 initial_setup.sql 脚本创建了一个非常基本的 CUSTOMER 表，该表作为本过程中的示例。 最好避免使用模式所有者身份来访问数据.而是应一个不同的帐户(如 AppSvr),该帐户由所有客户端连接共享，并由 Oracle 应用服务器处理。AppSvr 数据库用户不拥有任何对象，并且只拥有 CREATE SESSION 系统权限，但拥有对所有包含模式所有者(如 SHIP2004 模式的所有者)应用程序数据的表的 SELECT、INSERT、UPDATE 和 DELETE 权限。 支持文件中的 enable_connection.sql 脚本创建一个通常由运行在 Oracle 应用服务器上的应用程序使用的用户(如上所述)。 安全性实施 为实现所述的安全目标，除非您“授权”了连接(由在预定的 IP 地址处运行的 Oracle 应用服务器启用)，我们将使用一个数据库策略来隐藏 CUSTOMER 表中的记录，。此策略在安全管理器用户(如 Sec_Manager)下实现，因此即使从 SHIP2004 或 AppSvr 模式中也看不到它。

确定要使用的环境变量以及要由安全谓词检查的特定值是实现的问题。大量的潜在组合和特殊的网站详细信息将创建重要的入侵尝试障碍。 为安全实现中使用的所有定义创建一个没有任何权限(甚至是 CONNECT)的单独模式(如 Sec_Manager)作为占位符是比较可取的。所有对象将由 Sec_Manager 模式中的数据库管理员帐户创建。由于没有权限，此用户名甚至无法用于登录到数据库，因此所拥有的安全性定义将得到可靠地保护。(任何人甚至看不到与安全性相关的对象的定义。) 但本文最初的目标之一是为几个维护和支持人员成员实现 SQL*Plus 级别的访问。此紧急访问需要一个“安全通道”，它可以被授权用户轻松记住，但由于太长而无法写入到桌面即时贴中(可由任何人看到)，这种由于所保留的口令数目所导致的不利情况。本示例使用 CLIENT_IDENTIFIER 环境变量，但它可以为您所选择的任何环境变量或环境变量组合。 create_setup.sql 脚本(位于支持文件中)演示了如何根据以上描述创建安全实现模式、谓词函数以及安全策略。它还生成了几个数据列表，并使用不同的数据库登录权限演示了将在 CUSTOMER 表中看到(或看不到)的不同连接。它还演示了如何使用 dbms_session.set_identifier 函数进行解密，以通过 SQL*Plus 连接访问数据。