Oracle DBA 手记 4

数据安全警示录

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

未雨绸缪，防患未然

《Oracle DBA 手记 4》序言

在数据库领域十几年，我发现在国内技术人员往往一直在充当救火员的角色，企业常常也认为只有能够力挽狂澜、起死回生的技术人员，才是好的技术人员。而实际上，能够不犯错误、少犯错误，提前预防、规避灾难的技术人员才是企业技术环境的最有力保障,能够未雨绸缪，防患于未然才是更好的技术实践。

我们每年都帮助很多企业挽救数据、拯救危机，触发这本书写作的契机正是来自 2011 年 12 月 30 日和 31 日，连续的两个整天，从凌晨再到凌晨，连续挽救了几个用户的数据库灾难，这些灾难发生的那么简单，那么不可思议，在迈入 2012 这个神秘年份的一刻，深深的触动了我，我想，如果将这些案例描述出来，就可能帮助一些用户警醒借鉴，避免再陷入这样的困境。而**从别人的挫折中学习、借鉴，进而在自我的环境中未雨绸缪，防患于未然，是每个数据库管理人员和企业数据环境管理者应该具备的素质**。

写作这本书还和 2011 年底众多席卷而来的密码泄露事件有关，当我注视着最常用的几个密码都在互联网上被公开时，除了手忙脚乱的在各大网站修改密码，剩下的就是深深的遗憾。几乎所有从事 IT 行业的人，都深知安全的重要性，可是放在实际执行中，大家又往往习惯性失明，忽视了自己周围本应能够达到的力所能及之安全，很多专业人士就以这样或者那样的侥幸心理放任了风险的存在，并一步一步走向了安全危机。

对于数据库安全来说，通常我们认为缺乏的并非技术手段，更多的是缺乏规范和安全认知，如果用户都能够严格的遵循安全守则并应用现有的安全技术手段，数据库的安全性就能够大幅增强，我们的安全事故率也会大大降低。

所以我决定动笔，写下自己多年来所遭遇到的安全案例以及对于数据安全的思考，如果本书中的内容能够帮助一些企业规避错误，保全数据，挽救一些技术人员的时间，那么我将感到无比欣喜，于我们的生命中，最为宝贵的就是时间，寸金难买寸光阴。

信息安全

在传统的信息安全领域，存在三个基本的安全要素，这三个要素分别是：

保密性(Confidentiality)、**完整性**(Integrity)**和可用性**(Availability)，**缩写为** CIA。

这三个要素的基本定义如下：

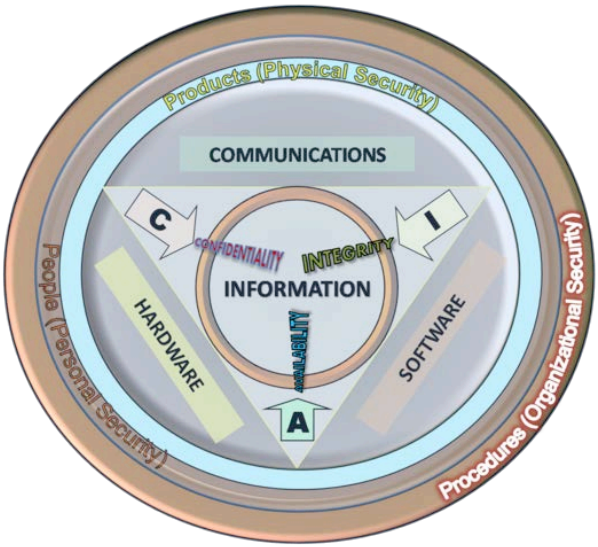
保密性（Confidentiality）：指信息在存储、使用和传输过程中的保密性，要确保信息在这些环节中不会泄漏给非授权方；

完整性（Integrity）：指信息在存储、使用和传输过程中不会被非授权用户篡改、变更，同时需要防止授权用户对系统及信息进行非授权篡改，保持信息在整个过程中内外的一致性；

可用性（Availability）：信息系统因其服务使命，必须在用户需要时，可以被正常访问，授权用户或实体对信息系统的正常使用不应被异常拒绝或中断，应当允许其可靠、及时地访问和获取信息及资源。高可用系统要求所有时间可用，要确保系统不因电源故障、硬件故障和系统升级等因素影响服务的可用性。

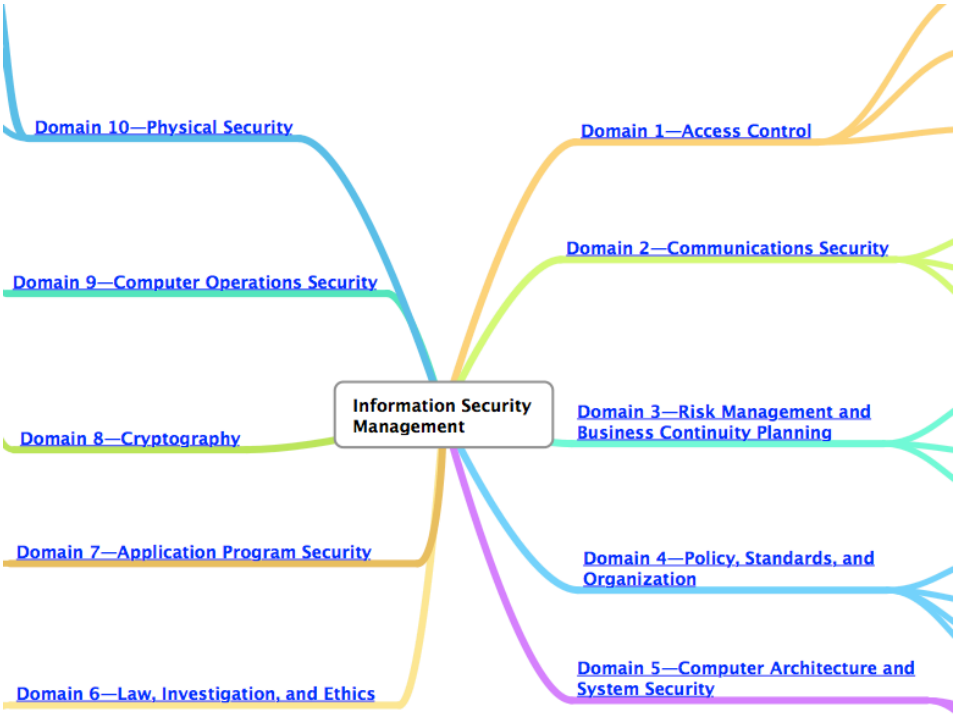
信息安全的三要素是对安全的概括和提炼，不同机构和组织，因为需求不同，对 CIA 的侧重也会有所不同，随着信息安全的发展，CIA 经过细化和补充，增加了许多新的内容，包括可追溯性（Accountability）、抗抵赖性（Non-repudiation）、真实性（Authenticity）、可控性（Controllable）等；与 CIA 三元组相反的一个概念是 DAD 三元组，即泄漏（Disclosure）、篡改（Alteration） 和破坏（Destruction），实际上 DAD 就是信息安全面临的最普遍的三类风险，是信息安全实践活动最终应该解决的问题。

CIA 的核心三要素涉及软件（Software）、硬件（Hardware）和通信（Communications）三个方面，下图清晰的描述了信息安全三要素及相关领域范畴：



信息安全的三要素（Information security From Wikipedia）

从 CIA 理念出发，通过对信息安全范畴所有相关主题精炼整理得到了一个标准化的知识体系，被称为公共知识体系—CBK（Common Body of Knowledge），CBK 包括 10 个知识范畴（Domain），对安全进行了全面的概括，具有极强的指导意义，下图对 10 领域进行了简单的列举（不同出版物描述略有不同）：



CBK 10 Doamin（参考 Handbook of Information Security Management）

以上 10 个范畴分别为：访问控制，电信、网络和互联网安全，风险管理和商务连续性计划，策略、标准和组织，计算机架构和系统安全，法律、调查和道德，应用程序安全，密码学，计算机操作安全，物理安全。

数据安全

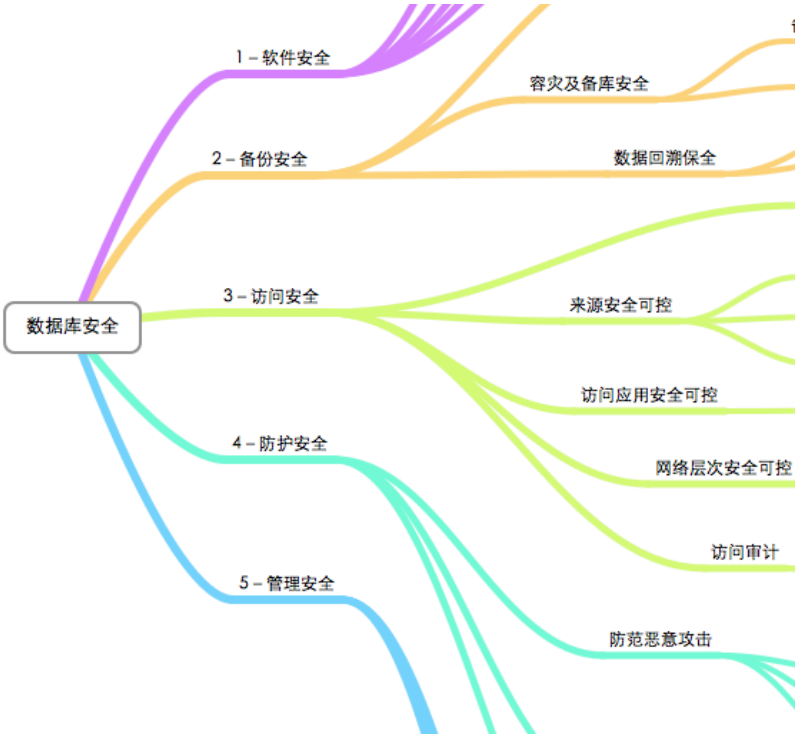
信息安全的核心是数据安全。

在数据安全的范畴内，也包含信息安全的诸多方面，根据多年的服务经验与思考，我们将安全划分为五大

方面，分别是：

软件安全、备份安全、访问安全、防护安全、管理安全

这五大方面是信息安全在数据领域的引申和映射，在企业数据安全中，这五大方面是相辅相成、互有交叉、共同存在的，下图是关于安全的一张思维导图，本书内容的案例涉及到了这五大方面：



在这五大安全方向中，可能出现**两种性质的安全问题，第一，由于内部管理不善而导致的数据安全问题；第二，由于外部恶意攻击入侵所带来的安全安问题**。通常我们把安全问题狭义化为后者，这实际上是片面的，在数据安全问题上，前者造成的数据损失、数据损毁，其发生率和影响度都远远超过后者。

下面我们对数据安全的五大方面做一下简要的分析和探讨：

1 .**软件安全是指我们选择的数据库产品、版本是否稳定安全；厂商所能提供的补丁集和 BUG 修正是否及时、基础硬件与操作系统是否经过认证**。很多用户在部署数据库软件时，仅仅选择了最容易获得的初始发布版本（如 Oracle Database 10.2.0.1 或者 Oracle Database 11.2.0.1 等），遗漏了可能已经存在的补丁修正，并且在运行维护中并不能够及时跟踪软件更新，也无法获得 BUG 信息、补丁修正和安全告警，这就使得软件本身的很多风险隐患得不到修正。如果软件安全无法保证，数据库安全的基础也就丧失了。

2 .**备份安全是指用户数据能否得到及时有效的备份保全，能否在故障灾难之后获得及时的恢复和挽救**。在数据库运行期，最为重要的就是备份安全，如果没有可靠的备份，将数据集中起来就只能是等待数据灾难，所以我们将备份安全提升到核心地位，备份以及随之衍生的容灾安全等，都是企业整体数据架构应该考虑的因素。很多企业在数据灾难之后因为缺乏有效备份而一蹶不振，根据 Gartner 2007 年的一份调查报告显示,在经历了数据完全丢失而导致系统停运的企业中，有 2/5 再也没能恢复运营，余下的企业也有 1/3 在两年内宣告破产,由此可见，由于备份安全问题导致的企业伤害可能远远大于黑客攻击。

3 .**访问安全是指用户数据库的访问来源和访问方式是否安全可控**。通常数据库系统处于 IT 系统的核心，其安全架构涉及主机、系统、存储、网络等诸多方面，如果没有明确的访问控制，缺乏足够的访问分析与管理，那么数据库的安全将是混乱和无法控制的。在应用软件使用和访问数据库时，要正确设置权限，控制可靠的访问来源，保证数据库的访问安全，唯有保证访问安全才能够确保数据不被越权使用、不被误操作所损害，通常最基本的访问安全要实现程序控制、网络隔离、来源约束等。

4 .**安全防范是指通过主动的安全手段对数据库通讯、传输等进行增强、监控、防护、屏蔽或阻断，诸如数据加密、审计、数据防火墙等技术都在这一范畴之内**。我们必须认识到，在 IT 技术高度发展的今天，风险是无处不在、层出不穷的，可能我们从未思考过的安全问题，每天都在不断涌现，所以在数据库环境中采取主动式防护，可以帮助我们监控分析和屏蔽很多未知风险，已经有很多成熟的产品和技术可以用于安全防范。

5 .**管理安全是指在企业数据的日常管理维护范畴内，能否充分保证数据安全以及服务的高可用连续提供**。诸如 DBA 的维护、文件的管理、参数或数据结构的变更等等都可能引入数据风险，管理安全要求我们通过规范、制度以及技术手段去确保维护管理安全；另外，基于硬件、电力等基础平台的故障都可能影响数据库服务的高可用性，在管理中要通过监控手段及时预警，通过集群、备库等切换与服务分担保障服务的连续性。

这就是数据安全的五大方面。

业界安全事故

在 2011 年的新闻报道中，我们注意到很多企业遭受到了严重的安全事故，影响深远，以下摘录了几起广为人知的数据安全事故，让我们一起看一看安全问题都出现在何处：

1. **陕西移动近 1400 万条个人信息遭泄露**

根据新闻报道（案件发生大约时间为 2011 年 3 月），犯罪嫌疑人所在的某技术公司承担着为陕西某电信企业提供手机资费计算系统软件平台开发、运行、维护、咨询、防毒等多项工作，可以获取该电信运营商手机用户号码、姓名、年龄、性别、身份证号、住址、每月通讯费用等资料。

犯罪嫌疑人为了个人利益，窃取用户信息并出售：

“朋友。。。向我要西安、榆林、延安、渭南等六七个地市的移动手机每个月话费消费 20 元以上的信息，内容包括手机号码、月话费消费情况、办卡区域、机主性别、出生年月等，我同意了。第二天我**在单位**将电脑连接到**省移动公司数据库**中，提取了 1000 余万条信息，每个地市建立一个文件夹，存储到我的笔记本电脑中。。。”

2. **高阳捷迅工程师利用支付宝漏洞盗取 11 万**

2009 年间，支付宝公司开通话费支付业务，用户都可以通过购买手机充值卡、充入支付宝账户后进行网上购物，高阳公司负责这一话费充值系统的运行维护。即在支付宝与移动、联通、电信之间搭建平台，负责将支付宝用户购买的手机充值卡转变为支付宝账户的存款。

犯罪嫌疑人是负责这一系统维护的工程师，在 2010 年 1 月至 3 月间，**他利用了这个系统的漏洞**，多次通过互联网进入高阳公司系统数据库，调取用户充值失败而暂存于此的充值卡信息，然后将其转入自己在支付宝设立的 48 个账户和快钱设立的 31 个账户，共计 111650 元。

3. CSDN 600 **用户密码泄露事件**

2011 年 12 月 21 日，一组安全事件在国内引发了轰动，黑客在网上公开了 CSDN 网站用户数据库，包括 600 **余万个明文的注册邮箱账号和密码可能遭集中曝光**。事件发生之后，CSDN 相关网页更一度紧急关闭，以升级为由暂时关闭。

随后又暴露了一系列的密码安全事故，多家大型网站的用户信息遭泄露。

4. Putty **中文版后门事件**

据 2012 年 2 月 1 日消息，**中文版** putty **等** SSH **远程管理工具被曝出存在后门**，该后门会自动窃取管理员所输入的 SSH 用户名与口令，并将其发送至指定服务器上。根据分析，此次事件涉及到来自 putty.org.cn、putty.ws、winscp.cc 和 sshsecure.com 站点的中文版 putty、WinSCP、SSHSecure 和 sftp 等软件，而这些软件的英文版本不受影响。

5. **赛门铁克遭黑客“破门” 千万用户信息安全存疑**

北京时间 2012 年 02 月 07 日，一个容量达 1.2GB、标题为“赛门铁克的 pcAnywhere 源代码遭泄露”的文件出现在了 BT 网站，并开放提供下载。

赛门铁克确认，pcAnywhere **源代码已被公开发布。这是黑客组织** Anonymous **在过去几周中所声称已获取的** 2006 **版本产品源代码的一部分**。黑客曾经向赛门铁克索取 5 万美元。

赛门铁克在与黑客的电子邮件中表示，“我们将向你支付 5 万美元。但是，我们需要确保你在收到钱后不把源代码发布到互联网上。在起初的三个月中，我们将每月支付 2500 美元。我们将从下周开始向你支付这笔费用。在三个月结束之后，在我们支付余额前，你要让我们相信你已经销毁了源代码。我们相信你不会没完没

了地讨价还价。”

在经过了数周有关源代码证据及如何转账的谈判后，双方谈判破裂，交易未能完成。

很快，在微博上出现了这样的“段子”：悲剧 - 安全软件源代码被黑客偷了;喜剧 - 人家只勒索 5 万美元;悲剧 - 赛门铁克要求分期付款，谈崩了;喜剧 - 只好报警;悲剧 - 黑客发布源代码……

我们可以注意到，数据安全问题无处不在，从软件到数据库，从维护人员到黑客，损害安全的因素不是越来越少，而是越来越多。这些事件更警示我们，要不断提升数据安全，防止安全事故的发生。

Oracle 数据库安全

其实 Oracle 数据库自 1977 年肇始之初，就一直将安全置于首位，从强大的数据恢复机制，到不断增强的加密以及安全防范措施。“Oracle”这个名字就是来自于美国中央情报局投资的项目代码，而 CIA 也正是 Oracle 最早期的用户之一。

接触过 Oracle 数据库的人都应当熟悉一个类似如下图所示的错误“ORA-00942：表或视图不存在”，这个简单的错误提示，最初就是在 CIA 的要求之下作为一项安全防范设定的，这个提示的安全意义在于：**避免提供任何具体的实质性提示性信息，以预防黑客的攻击性尝试**。由此可见，安全防范可以从每一个细节入手，安全是一项全面整体的技术实现，并非孤立的存在。

```
SQL> select * from emp;
select * from emp
              *
ERROR at line 1:
ORA-00942: table or view does not exist
```

受密码事件影响，我们首先在此从 Oracle 数据库的密码机制上来稍微深入了解一下 Oracle 的加密机制。虽然我们知道早在 Oracle 数据库版本 8 的年代，就已经提供了强大丰富的数据库加密功能，但是直至今日，恐怕半数以上的数据库中，仍然存放着用户的明文密码，并且未采用任何数据库安全增强机制。**这也就是我认为最重要的安全问题：我们并不缺乏安全防范手段，但是缺乏对于安全风险的认知**。

诚然，我们对于安全的认识是随着不断出现的安全事故逐步增强的，但是希望大家都能够有计划的逐步增强对于数据库的安全防范，**主动规划推进数据安全与从挫折中学习提高实有天壤之别**。对于 2011 年底的密码泄露事件，如果各大网站能够采取基本的技术手段对用户密码进行一定的加密，那么这次密码泄露的安全事件就不会显得那么初级和惹人恐慌，想一想明文密码和 MD5 加密串的区别？前者基本上意味着数据库从未从安全角度进行过任何思考和增强。

Oracle 数据库的用户信息及密码存储于一个名为 USER$的数据表中（所有者为 SYS 用户），我们可以通过基于 USER$表建立的 DBA_USERS 视图来查询和获得这些信息，包括加密的口令串。

在 Oracle Database 11g 之前，用户口令通过 DES 加密算法进行加密，使用用户名作为“Salt”，密码最长为 30 个字符，所有字母被强制转换为大写。从 Oracle 7 至 Oracle 10g，加密一直使用 username 和 password 串连之后进行 HASH 运算，例如 sys/temp1 和 system/p1 将会获得相同的 HASH 加密输出。

从 Oracle Database 11g 开始，Oracle 允许最多使用 30 个字符、大小写混合方式作为密码，同时支持 DES 和 SHA-1 算法进行加密（SHA-1 算法支持大小写混合，通过初始化参数 SEC_CASE_SENSITIVE_LOGON 开关），使用 password||salt 的方式进行 HASH 加密。

以下是 Oracle 9i 数据库中口令的加密形式，DBA_USERS 视图的 PASSWORD 字段显示了加密后的密钥：

```
SQL> select username,password from dba_users
  2  where username in ('SYS','SYSTEM','EYGLE');
USERNAME                       PASSWORD
------------------------------ ------------------------------
EYGLE                          B726E09FE21F8E83
SYSTEM                         A204A4CEB2C2F2FB
SYS                            7ABF43E21B3DD9A3
```

在 Oracle 11g 中，密码从 DBA_USERS 视图中隐藏起来，这进一步的增强了安全性，即便具有访问视图权限的用户，也无法获得口令的加密串，由此我们也可以看出 Oracle 数据库软件的安全增强历程：

```
SQL> select username,password from dba_users
  2  where username in ('SYS','SYSTEM','EYGLE');

USERNAME                       PASSWORD
------------------------------ ------------------------------
EYGLE
SYS
SYSTEM
```

口令的加密内容存储在底层的核心表（USER$是 Oracle 数据库的元数据表之一）中，以下 PASSWORD 字段存储的是 DES 加密值，SPARE4 存储的是 SHA-1 加密信息：

```
SQL> select * from v$version where rownum <2;

BANNER
--------------------------------------------------------------------------------
Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production

SQL> select name,password,spare4 from user$
  2  where name in ('SYS','SYSTEM','EYGLE');

NAME       PASSWORD         SPARE4
---------- ---------------- ------------------------------------------------------------
SYS        8A8F025737A9097A S:BBEFCBB86319E6A40372B9584DBCCA6B015BFE0C7DDF5B9593FB618E0D80
SYSTEM     2D594E86F93B17A1 S:C576FB5A54D009440AC047827392215C673528067BC06659EC56E3178BAB
EYGLE      B726E09FE21F8E83 S:65857F36842AEE4470828E9BE630FEED90A67CEF0D2B40C9FE9B558F6B49
```

关于口令的维护，Oracle 支持各种约束性限制（通过 utlpwdmg.sql 脚本启用），诸如复杂程度、长度、有效期、失败登陆次数等等，通过这些增强，Oracle 的口令限制可以定制出非常稳固的安全解决方案，如果你从未接触和研究过这些手段，那么可能就说明你的数据库还缺乏足够的第一层的安全防守。

如果我们能够从 Oracle 的安全策略入手，学习一下 Oracle 的口令安全解决方案，那么就能够构建一套较为完善的基本安全解决方案。从 Oracle 的第一个 Internet 版本 Oracle 8i（1998 年发布）开始，Oracle 就提供了一个加密包 DBMS_OBFUSCATION_TOOLKIT 用于数据安全防护，这个加密包支持 DES，3DES 和 MD5 加密算法。

通过非常简单的封装调用，DBMS_OBFUSCATION_TOOLKIT 包就能够实现数据加密，以下是一个简单的示例输出，对于给定字符串进行 MD5 加密，以 RAW 方式返回加密结果（通过创建稳固的函数，可以实现用户登陆时的即时加密、比较、认证）：

```
SQL> set serveroutput on
SQL> BEGIN
  2    dbms_output.put_line(utl_raw.cast_to_raw(
  3    DBMS_OBFUSCATION_TOOLKIT.MD5(INPUT_STRING =>'EYGLE')));
  4  END;
  5  /
F7FB70F1E13721034765EEC4EA1A3832

PL/SQL procedure successfully completed.
```

从 Oracle Database 10g 开始，DBMS_CRYPTO 包被引入到数据库中，该程序包支持更广泛的加密算法，并用于替代 DBMS_OBFUSCATION_TOOLKIT 包，在新的版本中，诸如 DES, 3DES, AES, RC4, MD5, SHA-1, MD4, HMAC_MD5, HMAC_SH1 等等加密算法和加密方式都被支持。

通过选定的加密算法和加密方式，可以对重要数据进行加密和解密，我们不仅可以实现对于密码或数值、字符数据的加密，甚至可以对类似 LOB 等非结构化数据进行加密。以下范例是使用 DES 算法 CBC 模式和 PKCS5 补码规则的加密解密实现，示例模拟对于信用卡卡号的处理过程，金融类企业数据的安全性更为突出，需要进

行安全加密的类型更为丰富：

```
SQL> set serveroutput on
SQL> DECLARE
  2    vcardno VARCHAR2(19) := '8610-1391-1812-8031';
  3    vcadraw RAW(128)     := utl_raw.cast_to_raw(vcardno);
  4    crawkey RAW(128)     := utl_raw.cast_to_raw('oracle10g');
  5    encyraw RAW(2048);
  6    decyraw RAW(2048);
  7  BEGIN
  8    dbms_output.put_line('CardNo : ' || vcardno);
  9    encyraw := dbms_crypto.encrypt(vcadraw, dbms_crypto.des_cbc_pkcs5, crawkey);
 10    dbms_output.put_line('EncdNo : ' || RAWTOHEX(utl_raw.cast_to_raw(encyraw)));
 11    decyraw := dbms_crypto.decrypt(src => encyraw, typ => dbms_crypto.des_cbc_pkcs5, KEY => crawkey);
 12    dbms_output.put_line('DecyNo : ' || utl_raw.cast_to_varchar2(decyraw));
 13  END;
 14  /
CardNo : 8610-1391-1812-8031
EncdNo : 42313246313041343030324538303234394546363245453734453341373830363245423039433745434645434631423 4
DecyNo : 8610-1391-1812-8031

PL/SQL procedure successfully completed.
```

我想重申的是，对于不同的数据库产品，都存在足够成熟的安全实现手段，应用这些安全手段就能够实现对于数据的基本保护，对于我们技术人最重要的是：**认识和重视数据安全问题，并逐步推动企业或组织应用安全手段进行数据安全增强**。重视数据，保护数据，这是每一位技术人的共同使命！

本书使命

本书所描述的所有恢复以及安全案例全部确有其事，但是基于用户隐私，我们隐去了所有客户相关的信息，摘录的内容涉及用户判断的，全部进行了处理，但是时间、故障内容一切属实。多年来的灾难挽救为我积累了很多素材，所以写作这本书是一次回顾的旅程，以一条主线，将众多的灾难挽救过程串联起来。这本书中的很多案例，尚属首次批露，而你也许还会注意到，书中的某些技术细节和恢复方法至今你从未在其他地方看到。

本书中的很多案例恢复非常艰难，拯救过程也花费了大量的人力、物力和时间，我们也因此赢得了数百万的商业合同，但是从内心上，我们永远不希望用户陷入到这样的境地，所以我写作了这本书，以使得这些曾经惨痛的教训可以具备更广泛的借鉴意义。

基于这些想法，我对每个案例的发生过程进行了描述，并且提出了供大家警示的规避法则。所以，**我希望这是一本写给大家看的数据安全之书，不仅仅是给技术人员，更重要的是给企业数据管理者，如果不看这些案例，你也许永远不会理解数据库如何会遭遇到灭顶之灾，你也许永远无法理解为何千里之堤一朝溃于蚁穴**。

当然，这仍然是一本相当深入的技术之书，我将很多案例的详细拯救过程记录下来，包括一些相当深入的技术探讨，这些技术探讨一方面可以帮助读者加深对于 Oracle 数据库技术的认知，一方面又可以在你遇到类似案例时，做出同样的营救工作。

对于我自身来说，年纪越长，就越是认识到这个世界上最为宝贵的就是时间，而如果我的分享，从警示到技术，能够帮助大家规避错误，少犯错误，在恢复时不走弯路，快速拯救数据，节省工程师们和用户的时间，那么这就是我最深刻之所愿，拯救时间，即为功德，这世界上，寸金永远也买不到一寸光阴。

这本书是用他人的灾难，为大家做警示，但愿我们都能够从中吸取教训，永远不要遇到本书所描述的种种灾难。

致谢

感谢我的朋友们，他们对我的帮助和支持使得本书的很多内容得以成型，刘磊在 ACOUG 上的演讲《猜测的力量》帮助我深入了关于 REDO 分析的案例；本书还引用了杨廷琨分析解决的一个 ASM 故障案例，此外，附录中老杨提供的函数对我们非常有用；感谢用户，是他们的信赖使得我们能够接触种种艰难的案例，并帮助他们挽回数据；感谢支持帮助过我的朋友们以及一贯支持我的读者们，这书中真挚的内容就是我最好的回报.

感谢我的好友丁晓强同学，他在支付宝公司进行了多年的安全与运维体系的建设与管理工作，他基于实践而来的对于安全和运维的真知灼见给予了本书很多肯綮的建议，这些建议让我对于安全有了更加系统全面的理解，从而也对本书的架构做出了调整，虽然有很多想法没能在本书中体现，但是我相信，我仍然有机会在未来和大家继续分享我从他那里学来的宝贵经验。

再次感谢杨廷琨，他在本书出版之前，通读了全书书稿，细致到帮我修改一个敲错的字母，一个写错的汉字，这份细致认真让我钦佩得无以复加；他还帮助我增加了两个警示条目，来自于他感触最为深刻的技术经历。老杨的工位和我相邻，在工作中我随时的请教可以即刻得到他绝妙的提示，让我节省了大量的工作时间，这是达成本书的另外一个重要助力。

我还要感谢我的太太 Julia 和儿子，以及我的家人，我为写作这本书，牺牲了很多陪伴他们的时间，也正因为有他们的支持，我才能够不断的写作下去。

最后，我但愿书中这些看起来似乎遥远的故事，能够警醒某些你曾经似曾相识的操作，并且永远不要面对这样的灾难。

盖国强（Eygle）

2012-01-20　初稿

2012-03-01　定稿于北京

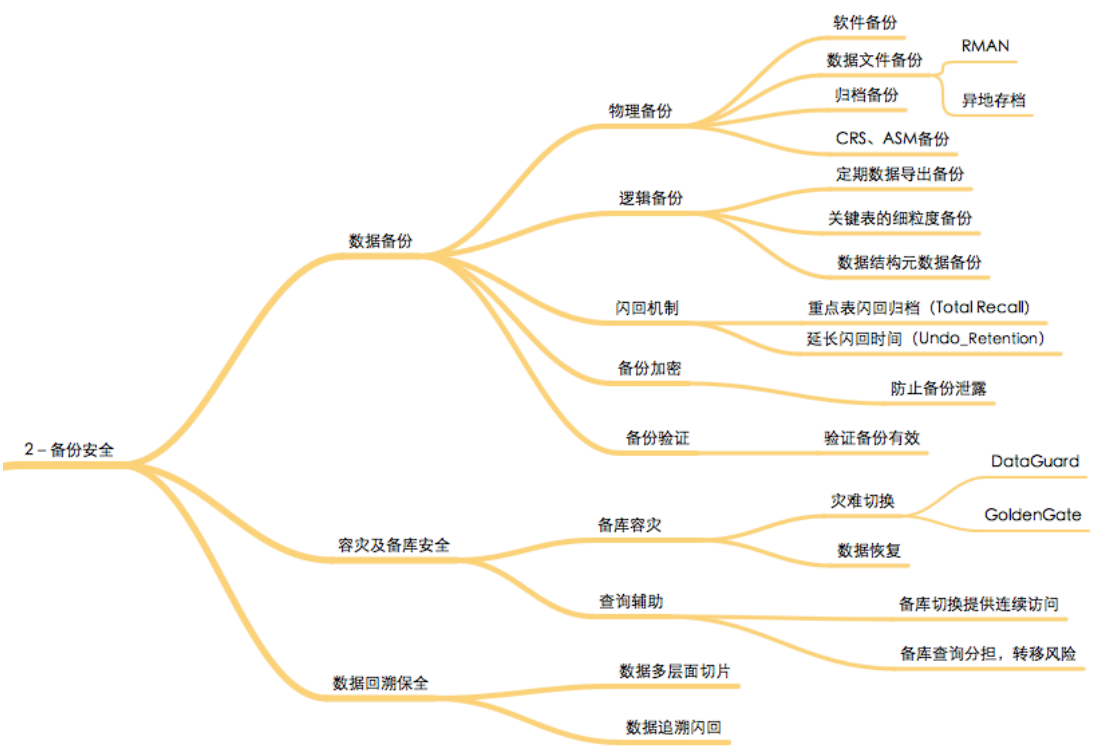
目　　录

靡不有初，鲜克有终

荡荡上帝，下民之辟。疾威上帝，其命多辟。

天生烝民，其命匪谌。靡不有初，鲜克有终。

——《诗经·大雅·荡》



“靡不有初，鲜克有终”，这句话的意思大致是说，人们做事大都能有一个良好的开端，但是很少有人能够善始善终。

每年岁末，我们都能够接触到大量的数据库安全相关案例，一个疏忽、一个原本为了安全而执行的操作，最终使数据库陷于绝境。而这些陷入绝境的数据库，多半都没能进行及时有效的备份，当灾难发生时，处理起来就非常棘手。

对于很多企业，备份缺失并非说明数据不够重要，很多时候客户会因为长期以来数据库一直稳定运行而麻痹大意，忽略或者忽视备份检查，直至出现故障时才发现备份不足，或已有的备份不可用，却已为时太晚。

而对于某些海量数据企业，常规的备份恢复时间已经不能满足业务连续性要求，所以一旦遭遇故障，某些时候可能仍然需要采取一些极端手段进行数据恢复和强制修复。

篇首图是关于备份安全的思考和总结，企业应当充分保障数据备份，并考虑容灾与备库的安全保障，对于特定的企业要求，可能还需要保证数据的可回溯性。所有这些都是在备份安全中应当考虑的。

关于图中内容，还需要着重强调以下几点。

1. 注意备份加密的重要性。如果不能保护备份和备份集，那么一旦发生丢失、泄露就可能导致严重的安全问题。
2. 在数据库管理维护中，应当明确闪回机制的功能。在出现常规数据误操作如删除等问题时，可以优先尝试使用闪回功能进行恢复。
3. 定期的备份检查和备份验证必不可少。如果没有确认和验证，那么备份的有效性就无法保证，在故障发生时就很有可能遭遇无法恢复的灾难。
4. 备库是非常有力的数据保障机制。如果环境许可，对生产数据库构建一个 DataGuard 备库容灾环境，可以在危急关头挽救数据。

我们应当深入思考备份安全问题，如果能够构建有效的备份保全策略，那么企业的数据安全就有了基本的保障。

下面我们就来一起审视几则数据库安全事故，这些数据库在 2011/2012 岁末未能善终。

以空间之由

误操作删除数据文件恢复案例两则

我们都应当熟悉以下这段诗句，从一颗钉子，到一个国家。

少了一颗钉子，丢了一块蹄铁；
少了一块蹄铁，丢了一匹战马；
少了一匹战马，丢了一个骑手；
少了一个骑手，丢了一场胜利；
少了一场胜利，丢了一个国家。

——詹姆斯·格莱克《混沌学》

我们见过很多客户，因为空间紧张而东拼西凑，最终导致难以挽回的数据灾难，回头看来是那么得不偿失！**因为一块硬盘，损失整个数据库，看起来是多么的荒诞。**然而，这样荒诞的事情还在不断发生。

灾难描述

2011 年 12 月 30 日，某运营商客户，在遭受数据损失之后请求我们协助进行数据恢复。整个数据灾难的过程如下。

1. 凌晨，数据库归档日志写满磁盘，因而无法继续归档，数据库服务中断。
2. 在进行空间释放时，删除了一个认为不再需要的目录。
3. 删除目录之后，发现数据库服务受到影响。
4. 经确认，该目录包含 7 个 16GB 大小的在用数据文件。
5. 在试图通过备份来恢复时，发现之前的备份是不成功的。
6. 灾难形成。

这是一个由删除引发的严重故障，如果数据不可恢复，灾难影响将是非常严重的。

案例警示

分析整个灾难的形成过程后，我们总结出了如下教训。

1. **数据库需要全面的系统规划和监控。**

首先，客户的数据库系统缺乏监控，直到归档路径空间使用率达到 100%，出现错误影响了业务应用，客户才意识到数据库出现了问题。按照常规运维要求，系统应当部署必要的监控手段，在空间达到一定阈值时即报警，比如空间使用率达到 80%。

我曾经反复警告：不要轻信归档模式的好处，对于没有良好备份、运维的数据库系统，启用归档模式往往是个灾难，只会反复带来麻烦。而且，归档模式不可避免地要牺牲一定的性能。

此外，在规划数据库系统时，也应当尽可能预留足够的磁盘空间。如今磁盘的成本已经越来越低，因为磁盘空间而导致灾难实属不该。

2. **对数据库的破坏性操作需要谨慎。**

在出现空间告警后，立即清理空间显然过于草率。在操作系统上的空间清理应当和数据库操作遵循同样的守则，即：**在任何破坏性操作之前，应当进行反复确认并根据需要进行有效备份**。

相应的操作人员应当铭记：**情况越紧迫，越应冷静，避免犯下无法挽回的低级错误**。

在我们的职业生涯中，关于删除文件养成的一个习惯是：**以转移（MOVE）替代删除（DELETE/RM），经过一段时间观察确保无影响后，再进行最终的文件移除**。

删除，这个简单的操作导致了数不清的灾难，因此应当极为谨慎地对待。

3. **数据环境运维必须遵守一定的安全守则。**

在数据环境的运维中，应当制订制度性的规范并严格执行。规范和制度会增加工作的复杂性，但也会极大地避免出现误操作的概率。

无约束的技术自由，会带来无法预料的后果；良好的规范和制度，其良性输出是可以预期和预知的。

在本例的用户环境中，显然没有明确的维护规则，维护人员并不清楚空间的分配使用情况，也不清楚特定目录存储的内容和含义，这才发生了误删除操作。如果系统能够维护一份部署文档，明确各目录存储的用途，那么错误也就不会这么容易发生了。

按照正常的规则，数据库服务器应当维护一份文件部署分布文档，明确指出各目录结构及用途，然后制订空间清理规则；在发生空间紧张时，需要通过文档来确定哪些目录可以删除，哪些目录必须保留。这样就可以