```
Completed:
DROP TABLESPACE XF_PRODUCT_TEMP   INCLUDING CONTENTS AND DATAFILES CASCADE
CONSTRAINTS
```

接下来的日志输出显示操作人员完全不熟悉 ASM 技术。用户试图在 E 分区上创建数据文件，而该分区实际上是裸分区，未创建文件系统。这也说明未能有人审核相应的脚本。从下面的命令输出来看，这些命令语句应该是通过工具发出来的。也就是说这次维护操作根本没有明确的计划和步骤，临时发挥的现场操作导致了一片混乱。

以下创建尝试失败了。

```
Sat Dec 31 12:34:57 2011
CREATE SMALLFILE
    TABLESPACE "XF_PRODUCT"
    LOGGING    DATAFILE
    'E:\ORACLE\PRODUCT\10.2.0\ORADATA\ORCL\XF_PRODUCT01.DBF' SIZE 100M
    EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT  AUTO


ORA-1119 signalled during:
CREATE SMALLFILE
    TABLESPACE "XF_PRODUCT"
    LOGGING    DATAFILE
    'E:\ORACLE\PRODUCT\10.2.0\ORADATA\ORCL\XF_PRODUCT01.DBF' SIZE 100M
    EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT  AUTO
..
```

这里的 ORA-1119 错误的含义是文件创建失败，可能的原因是空间不足，本例中是因为该分区为裸分区。

```
[ora11g@ENMO ~]$ oerr ora 1119
01119, 00000, "error in creating database file '%s'"
// *Cause:  Usually due to not having enough space on the device.
// *Action:
```

接下来还有很多出错的创建命令被执行，比如以下创建表空间的脚本遇到了 ORA-1543 错误，该错误提示要创建的表空间已经存在。在用户的数据库中，这个表空间确实存在，不曾删除（这其实是幸运），所以也就无法再次创建。

```
CREATE SMALLFILE
```

```
    TABLESPACE "XF_PRODUCT_HISTORY"
    LOGGING    DATAFILE
    'E:\ORACLE\PRODUCT\10.2.0\ORADATA\ORCL\XF_PRODUCT_HISTORY01.DBF' SIZE 100M
    EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT  AUTO

ORA-1543 signalled during:
CREATE SMALLFILE
    TABLESPACE "XF_PRODUCT_HISTORY"
    LOGGING    DATAFILE
    'E:\ORACLE\PRODUCT\10.2.0\ORADATA\ORCL\XF_PRODUCT_HISTORY01.DBF' SIZE 100M
    EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT  AUTO
..
```

这些操作也充分说明了用户及现场操作人员并不清楚自己在做什么，以及这样做可能产生的严重后果。

用户最后的正确选择是没有继续执行其他操作，而是直接寻求技术支持。对于这种情况，虽然文件从数据库中被删除，在 ASM 上也不可见，但是在存储级别的块级仍然存在，只要没有被覆盖，就仍然可以恢复出来。

ASM 默认以 1M 的单元（AU，ASM Unit）为单位进行数据分片存储。于是我们通过存储恢复软件扫描存储，找到这些 1M 大小的数据片，再进行整合拼接，恢复了 3 个数据库文件。然后结合用户之前备份的表结构信息，通过 Oracle 的内部数据抽取工具 DUL 将文件中的数据提取出来，加回到数据库之中，完成了数据恢复。

在 ASM 中创建表空间，可以极其简单。

```
SQL> create tablespace XF_PRODUCT datafile size 500M ;
表空间已创建。
```

这样才算利用到了 ASM 的便利之处。

在数据导入时，我们首先找到用户及口令信息，修改一个临时口令。

```
SQL> select username,password from dba_users where username='YNXF_PRODUCT';
USERNAME                       PASSWORD
------------------------------ ------------------------------
YNXF_PRODUCT                   58C18F637CFB1EED
SQL> alter user ynxf_product identified by yn;
用户已更改。
```

此后，利用之前的备份导入数据结构。

```
C:\>imp ynxf_product/yn file=ynxf_product100111.dmp rows=n ignore=yes full=yes log=prod.log
连接到: Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

经由常规路径由 EXPORT:V10.02.01 创建的导出文件
已经完成 ZHS16GBK 字符集和 AL16UTF16 NCHAR 字符集中的导入
. 正在将 YNXF_PRODUCT 的对象导入到 YNXF_PRODUCT
IMP-00017: 由于 ORACLE 错误 27477, 以下语句失败:
 "BEGIN  "
 "dbms_scheduler.create_job('"JOB_CLEAR_PRODUCT_DATA"',"
 "job_type=>'STORED_PROCEDURE', job_action=>"
.....
IMP-00091: 下列函数和对象出现以上错误: CREATE JOB_CLEAR_PRODUCT_DATA。将跳过此对
象的其余 PL/SQL 块。
即将启用约束条件...
成功终止导入, 但出现警告。
```

然后通过恢复的 DMP 文件导入数据。由于存在依赖关系，需要按照单表方式分先后进行导入，以下是个别的导入范例。

```
C:\>imp ynxf_product/yn file=ynxf10.dmp ignore=yes log=imp.log statistics=none tables=T_CHECKITEM
commit=yes
经由常规路径由 EXPORT:V10.02.01 创建的导出文件
警告: 这些对象由 YNXF 导出, 而不是当前用户

已经完成 ZHS16GBK 字符集和 AL16UTF16 NCHAR 字符集中的导入
. 正在将 YNXF 的对象导入到 YNXF_PRODUCT
. 正在将 YNXF 的对象导入到 YNXF_PRODUCT
. . 正在导入表                      "T_CHECKITEM"导入了       362418 行
成功终止导入, 没有出现警告。
```

最后将用户口令修改为原始加密串，完成恢复。

```
SQL> alter user ynxf_product identified by values '58C18F637CFB1EED';
用户已更改。
SQL> select username,password from dba_users where username='YNXF_PRODUCT';
```

```
USERNAME                       PASSWORD
------------------------------ ------------------------------
YNXF_PRODUCT                   58C18F637CFB1EED
```

这个案例的恢复得益于最后创建的表空间未写入 ASM 磁盘组。如果写入则原有被删除的文件就可能被覆盖，那样就不可能实现完全恢复了。

对于前面提到的第二个案例，导入文件损坏，在导入时遇到如下错误。

```
IMP-00009: abnormal end of export file
```

中文版提示大致如下。

```
IMP-00009: 导出文件异常结束
IMP-00028: 上一个表的部分导入已回退: 回退 29050 行
```

这种情况通常是因为导出文件异常，当导入最后发现异常时，会回退所有操作。

回退在数据库处理是正常的，因为导出文件异常可能导致局部数据丢失，使数据的一致性和完整性无法确保。如果想导入部分完好的数据，可以指定 commit=yes 参数，如此即可提交这些残存记录。

以安全之期

基于对数据安全、备份安全的期望，我们认为对于备份有效性的验证、备份加密都非常重要，以下简单介绍一下与此相关的技术内容。

VALIDATE 实现备份验证

RMAN 工具提供了 VALIDATE 命令，可用于校验备份集的有效性，常用命令如下。

```
restore validate controlfile;
restore validate spfile;
restore validate database;
restore validate archivelog between sequence_low and sequence_high;
```

验证命令会检查备份的存在性、完好性和可恢复性，帮助我们确认备份有效与否，但是并不会进行实际的恢复动作。

比如验证控制文件和参数文件。

```
RMAN> restore validate controlfile;
Starting restore at 25-JAN-10
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=388 devtype=DISK

channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
    backup piece /opt/oracle/product/db10g/dbs/c-1341966532-20100125-01
channel ORA_DISK_1: restored backup piece 1
piece handle=/opt/oracle/product/db10g/dbs/c-1341966532-20100125-01
channel ORA_DISK_1: validation complete, elapsed time: 00:00:02
```

```
Finished restore at 25-JAN-10
RMAN> restore validate spfile;
Starting restore at 25-JAN-10
using channel ORA_DISK_1

channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
        backup piece /opt/oracle/product/db10g/dbs/c-1341966532-20100116-00
channel ORA_DISK_1: restored backup piece 1
piece handle=/opt/oracle/product/db10g/dbs/c-1341966532-20100116-00
channel ORA_DISK_1: validation complete, elapsed time: 00:00:02
Finished restore at 25-JAN-10
```

根据备份集的大小，验证全备份的过程会有所不同，通常需要较长的时间。以下是一个全备份的验证过程。

```
RMAN> restore validate database;
Starting restore at 25-JAN-10
using channel ORA_DISK_1

data file 22 will be created automatically during restore operation
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
          backup piece /data3/ordrbak/orderfullback_order_20100124_4691
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4691 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:02:36
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
         backup piece /data3/ordrbak/orderfullback_order_20100124_4692
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4692 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:01:45
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
         backup piece /data3/ordrbak/orderfullback_order_20100124_4693
```

```
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4693 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:00:26
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
         backup piece /data3/ordrbak/orderfullback_order_20100124_4694
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4694 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:00:56
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
          backup piece /data3/ordrbak/orderfullback_order_20100124_4695
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4695 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:00:07
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
         backup piece /data3/ordrbak/orderfullback_order_20100124_4696
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4696 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:00:04
channel ORA_DISK_1: starting validation of datafile backupset
channel ORA_DISK_1: reading from
          backup piece /data3/ordrbak/orderfullback_order_20100124_4697
channel ORA_DISK_1: restored backup piece 1
piece handle=/data3/ordrbak/orderfullback_order_20100124_4697 tag=order
channel ORA_DISK_1: validation complete, elapsed time: 00:00:07
failover to previous backup

data file 22 will be created automatically during restore operation
Finished restore at 25-JAN-10
```

验证命令并不会真正执行恢复，所以可以免去异机测试的麻烦，同时又可以检查备份的有效性，是应当定期执行的操作。

数据库备份加密

从 10g R2 开始，Oracle 数据库提供了备份加密功能。通过加密，可以保护备份文件，防止因为备份泄露导致的数据安全问题。

在 RMAN 中，可以查询数据库的备份加密算法和备份加密设置。以下显示数据库的加密算法为 AES128，当前未启用备份加密。

```
RMAN> show encryption for database;
RMAN configuration parameters for database with db_unique_name ORCL are:
CONFIGURE ENCRYPTION FOR DATABASE OFF; # default
RMAN> show encryption algorithm;
RMAN configuration parameters for database with db_unique_name ORCL are:
CONFIGURE ENCRYPTION ALGORITHM 'AES128'; # default
```

通过视图 v$RMAN_ENCRYPTION_ALGORITHMS 可以查询数据库支持的加密算法。以下是来自 Oracle Database 11g R2 的查询输出，Oracle 10g 支持的加密算法与此相同。

```
SQL> select * from v$RMAN_ENCRYPTION_ALGORITHMS;
ALGORITHM_ID ALGORITHM_NAME                   ALGORITHM_DESCRIPTION          IS_ RES
------------ -------------------------------- ------------------------------ --- ---
           1 AES128                           AES 128-bit key                YES NO
           2 AES192                           AES 192-bit key                NO  NO
           3 AES256                           AES 256-bit key                NO  NO
```

RMAN 的这些选项可通过 CONFIGURE 命令进行修改。如以下命令修改了加密算法。

```
RMAN> configure encryption algorithm 'AES256';
new RMAN configuration parameters:
CONFIGURE ENCRYPTION ALGORITHM 'AES256';
new RMAN configuration parameters are successfully stored
```

RMAN 的加密支持三种模式，即口令模式、透明模式（通过 Wallet 本地钱包加密）和混合模式。通常基于 Wallet 的加密方式因其透明性而更加安全，口令模式因引入口令而增加了额外的风险，所以如果不需要转移备份，通常不建议使用口令加密模式。

以下分别介绍这三种加密模式。

口令模式

口令模式是最为简单的备份加密模式。备份的时候通过以下语句设置备份密码，然后备份数据库或对应的表空间、数据文件等。

```
solaris*orcl-/home/oracle$ rman target /
Recovery Manager: Release 11.2.0.3.0 - Production on Wed Feb 15 10:58:19 2012
Copyright (c) 1982, 2011, Oracle and/or its affiliates.  All rights reserved.

connected to target database: ORCL (DBID=1299676637)
RMAN> set encryption on identified by "eygle" only;
executing command: SET encryption
using target database control file instead of recovery catalog
RMAN> backup database;
Starting backup at 15-FEB-12
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=179 device type=DISK
channel ORA_DISK_1: starting full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
input datafile file number=00004 name=/u01/app/oracle/oradata/ORCL/users01.dbf
input datafile file number=00003 name=/u01/app/oracle/oradata/ORCL/undotbs01.dbf
input datafile file number=00002 name=/u01/app/oracle/oradata/ORCL/sysaux.dbf
input datafile file number=00001 name=/u01/app/oracle/oradata/ORCL/system01.dbf
channel ORA_DISK_1: starting piece 1 at 15-FEB-12
channel ORA_DISK_1: finished piece 1 at 15-FEB-12
piece handle=/ORCL/backupset/2012_02_15/o1_mf_nnndf_TAG20120215T105827_7mp7tnfh_.bkp
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:55
channel ORA_DISK_1: starting full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
including current control file in backup set
including current SPFILE in backup set
channel ORA_DISK_1: starting piece 1 at 15-FEB-12
channel ORA_DISK_1: finished piece 1 at 15-FEB-12
piece handle /ORCL/backupset/2012_02_15/o1_mf_ncsnf_TAG20120215T105827_7mp7wd9y_.bkp
```

```
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:01
Finished backup at 15-FEB-12
```

如果恢复时不提供密码，则会提示错误，不能解密，恢复不能执行。

```
RMAN> startup mount;
Oracle instance started
database mounted

Total System Global Area     835104768 bytes

Fixed Size                     2230592 bytes
Variable Size                620758720 bytes
Database Buffers             209715200 bytes
Redo Buffers                   2400256 bytes
RMAN> restore database;
Starting restore at 15-FEB-12
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=170 device type=DISK

channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
RMAN-00571: ===========================================================
RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============
RMAN-00571: ===========================================================
RMAN-03002: failure of restore command at 02/15/2012 11:03:42
ORA-19870: error while restoring backup piece
    /ORCL/backupset/2012_02_15/o1_mf_nnndf_TAG20120215T105827_7mp7tnfh_.bkp
ORA-19913: unable to decrypt backup
ORA-28365: wallet is not open
```

恢复时，需要指定解密的密码才可以恢复。以下是指定密码之后的恢复过程。

```
RMAN> set decryption identified by "eygle";
executing command: SET decryption
```

```
RMAN> restore database;
Starting restore at 15-FEB-12
using channel ORA_DISK_1

channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
channel ORA_DISK_1: piece
handle=backupset/2012_02_15/o1_mf_nnndf_TAG20120215T105827_7mp7tnfh_.bkp
channel ORA_DISK_1: restored backup piece 1
channel ORA_DISK_1: restore complete, elapsed time: 00:01:55
Finished restore at 15-FEB-12
```

透明模式

透明模式依赖于本地配置的 Wallet（钱包），无须设置密码。如果备份仅在本地进行维护，建议采用这样的加密方法，更安全可靠，更简便。这种方式需要额外配置 Oracle Encryption Wallet。

Oracle Encryption Wallet 配置的简要步骤如下。

配置 SQLNET.ORA 参数文件

设置 Wallet 地址信息，目录决定了创建 Wallet 钱包的地址。

```
solaris*orcl-/home/oracle$ cd $ORACLE_HOME/network/admin
solaris*orcl-/u01/app/oracle/product/11.2.0/db_1/network/admin$ cat sqlnet.ora
ENCRYPTION_WALLET_LOCATION=(SOURCE=(METHOD=FILE)(METHOD_DATA=(DIRECTORY=/u01/app/orac
le)))
```

创建 Wallet

在 SYS 用户下创建 Wallet。

```
SQL> alter system set encryption key authenticated BY "eygle";
System altered.
SQL> ! ls -l /u01/app/oracle/ewallet*
```

```
-rw-r--r--   1 oracle   dba         1573  2 月 15 日 12:37 /u01/app/oracle/ewallet.p12
```

创建之后 Wallet 自动打开，但在此后重启数据库后需要手工开启。

关闭和开启 Wallet 的命令大致如下。

```
SQL> alter system set encryption wallet close identified by "eygle";
System altered.
SQL> alter system set wallet open identified by "eygle";
System altered.
```

接下来就可以使用 Wallet 实现透明的数据加密。以下是加密备份的过程示范。

```
RMAN> configure encryption for database on;
old RMAN configuration parameters:
CONFIGURE ENCRYPTION FOR DATABASE OFF;
new RMAN configuration parameters:
CONFIGURE ENCRYPTION FOR DATABASE ON;
new RMAN configuration parameters are successfully stored
RMAN> set encryption on;
executing command: SET encryption
RMAN> backup database;
Starting backup at 15-FEB-12
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=180 device type=DISK
channel ORA_DISK_1: starting full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
input datafile file number=00004 name=/u01/app/oracle/oradata/ORCL/users01.dbf
input datafile file number=00003 name=/u01/app/oracle/oradata/ORCL/undotbs01.dbf
input datafile file number=00002 name=/u01/app/oracle/oradata/ORCL/sysaux.dbf
input datafile file number=00001 name=/u01/app/oracle/oradata/ORCL/system01.dbf
channel ORA_DISK_1: starting piece 1 at 15-FEB-12
channel ORA_DISK_1: finished piece 1 at 15-FEB-12
piece handle=/backupset/2012_02_15/o1_mf_nnndf_TAG20120215T124518_7mpg2z1f_.bkp
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:35
channel ORA_DISK_1: starting full datafile backup set
channel ORA_DISK_1: specifying datafile(s) in backup set
```

```
including current control file in backup set
including current SPFILE in backup set
channel ORA_DISK_1: starting piece 1 at 15-FEB-12
channel ORA_DISK_1: finished piece 1 at 15-FEB-12
piece handle=/backupset/2012_02_15/o1_mf_ncsnf_TAG20120215T124518_7mpg4334_.bkp
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:01
Finished backup at 15-FEB-12
```

如果恢复时 Wallet 关闭，则恢复不可进行，会报告不可解密的错误。

```
SQL> alter system set encryption wallet close identified by "eygle";
System altered.
RMAN> restore database;
Starting restore at 15-FEB-12
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=170 device type=DISK

channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
ORA-19913: unable to decrypt backup
ORA-28365: wallet is not open
```

在打开 Wallet 的情形下，恢复可以正确执行。

```
RMAN> startup mount;
connected to target database (not started)
Oracle instance started
database mounted
RMAN> sql 'alter system set wallet open identified by "eygle"';
sql statement: alter system set wallet open identified by "eygle"
RMAN> restore database;
Starting restore at 15-FEB-12
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=170 device type=DISK
```

```
channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
channel ORA_DISK_1: restoring datafile 00001 to /oradata/ORCL/system01.dbf
channel ORA_DISK_1: restoring datafile 00002 to /oradata/ORCL/sysaux.dbf
channel ORA_DISK_1: restoring datafile 00003 to /oradata/ORCL/undotbs01.dbf
channel ORA_DISK_1: restoring datafile 00004 to /oradata/ORCL/users01.dbf
channel ORA_DISK_1: restored backup piece 1
channel ORA_DISK_1: restore complete, elapsed time: 00:01:45
Finished restore at 15-FEB-12
```

混合模式

混合模式是同时启用 Wallet 和密码：在本地模式下，不需要密码恢复，实现透明；在异地异机恢复时，则需要提供口令恢复。

混合模式的密码设置如下，省去了 only 关键字。

```
RMAN> set encryption on identified by "mypass";
```

在异机恢复时，只需要指定口令就可以恢复，不需要 Wallet 的配合。

```
RMAN> set decryption on identified by "mypass";
RMAN> restore database;
```

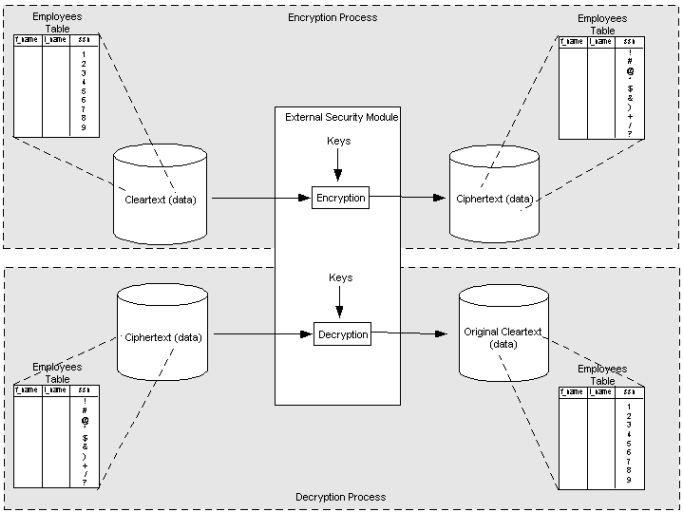
上述三种加密方式，可以根据需要选择使用。

透明加密（TDE）技术

RMAN 的透明加密，使用到了 Wallet 模式，这实际上是 Oracle 透明加密技术（Transparent Data Encryption）的一部分。

Oracle 的透明数据加密，是 Oracle 高级安全选项中的一个部分，需要额外支付软件费用。

这一选项，可以结合多种手段进行加密，包括使用 Wallet（PKCS#12 标准），以及支持 PKCS#11 RAS 硬件设备。在 Oracle 10g 中，透明加密支持基于列级的加密，而在 Oracle 11g R2 中，增加了基于表空间的透明加密。以下是官方文档中关于加密/解密的流程图。



以下是 Windows 上基于 TDE 的一个简单测试范例。

首先，在 SQLNET.ora 文件中增加如下一段。

```
ENCRYPTION_WALLET_LOCATION=
   (SOURCE=(METHOD=FILE)(METHOD_DATA=
      (DIRECTORY=D:\Oracle\11.2.0\NETWORK\ADMIN\encryption_wallet\)))
```

然后，在 SQL*Plus 中创建 Wallet 密钥。

```
SQL> connect / as sysdba
Connected.
SQL> alter system set encryption key authenticated by "eygle";
System altered.
```

关闭和打开 Wallet。

```
SQL> alter system set encryption wallet close;
alter system set encryption wallet close
*
ERROR at line 1:
ORA-28390: auto login wallet not open
SQL> alter system set encryption wallet close identified by "eygle";
System altered.
```

```
SQL> alter system set wallet open identified by "eygle";
System altered.
```

在创建数据表时可以指定加密。

```
SQL> connect eygle/eygle
Connected.
SQL> create table tde (id number(10),data varchar2(50) encrypt);
Table created.
SQL> insert into tde select user_id,username from dba_users;
9 rows created.
SQL> commit;
Commit complete.
SQL> connect / as sysdba
Connected.
SQL> select * from eygle.tde;
        ID DATA
---------- --------------------------------------------------
         0 SYS
         5 SYSTEM
        34 EYGLE
         9 OUTLN
        31 APPQOSSYS
        30 DBSNMP
        32 WMSYS
        14 DIP
        21 ORACLE_OCM
```

加密和解密是自动进行的。查询 dba_encrypted_columns 视图可以找到加密列。

```
SQL> select * from dba_encrypted_columns;
OWNER      TABLE_NAME      COLUMN_NAME          ENCRYPTION_ALG            SAL
INTEGRITY_AL
---------- --------------- -------------------- ------------------------- ---
------------
EYGLE      TDE             DATA                 AES 192 bits key          YES SHA-1
```

如果关闭 Wallet，则加密列不可访问。

```
SQL> select * from eygle.tde;
select * from eygle.tde
                    *
ERROR at line 1:
ORA-28365: wallet is not open
SQL> alter system set encryption wallet close identified by "eygle";
System altered.
SQL> select * from eygle.tde;
select * from eygle.tde
                    *
ERROR at line 1:
ORA-28365: wallet is not open
SQL> desc eygle.tde
 Name                                      Null?    Type
 ----------------------------------------- --------
------------------------------
 ID                                        NUMBER(10)
 DATA                                      VARCHAR2(50) ENCRYPT
SQL> select id from eygle.tde;
        ID
----------
         0
         5
        34
         9
        31
        30
        32
        14
        21

9 rows selected.
```

在加密列时，存在两个选项：Salt 和 No Salt。

Salt在加密前会对数据增加随机字符串,以增加破解的难度,使同样的字符串加密结果不同;而对于No Salt，则同样字符串可以获得同样的加密输出，其安全性相对略低。

在加密列上，如果使用 Salt 方式，则不能创建索引。Salt 加密和索引两种属性互斥，不能同时设置。

```
SQL> create index idx01 on tde(data);
create index idx01 on tde(data)
                           *
ERROR at line 1:
ORA-28338: Column(s) cannot be both indexed and encrypted with salt
```

而如果使用默认 Salt 方式加密，则允许对于加密列创建索引。

```
SQL> create table tde2 (id number(10) encrypt no salt,data varchar2(50) );
Table created.
SQL> insert into tde2 select user_id,username from dba_users;
9 rows created.
SQL> select * from tde2;
        ID DATA
---------- --------------------------------------------------
         0 SYS
         5 SYSTEM
        34 EYGLE
         9 OUTLN
        31 APPQOSSYS
        30 DBSNMP
        32 WMSYS
        14 DIP
        21 ORACLE_OCM

9 rows selected.
SQL> commit;
Commit complete.
SQL> create index idx1 on tde2(id);
Index created.
```

当执行导出时，Oracle 会给出提示。

```
D:\>expdp eygle/eygle directory=temp dumpfile=tde2.dmp tables=TDE
Export: Release 11.2.0.2.0 - Production on Thu Sep 8 15:35:19 2011
Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Starting "EYGLE"."SYS_EXPORT_TABLE_01":  eygle/******** directory=temp
dumpfile=tde2.dmp tables=TDE
Estimate in progress using BLOCKS method...
Processing object type TABLE_EXPORT/TABLE/TABLE_DATA
Total estimation using BLOCKS method: 64 KB
Processing object type TABLE_EXPORT/TABLE/TABLE
. . exported "EYGLE"."TDE"                               5.562 KB       9 rows
ORA-39173: Encrypted data has been stored unencrypted in dump file set.
Master table "EYGLE"."SYS_EXPORT_TABLE_01" successfully loaded/unloaded
******************************************************************************
Dump file set for EYGLE.SYS_EXPORT_TABLE_01 is:
  D:\TEMP\TDE2.DMP
Job "EYGLE"."SYS_EXPORT_TABLE_01" completed with 1 error(s) at 15:35:23
```

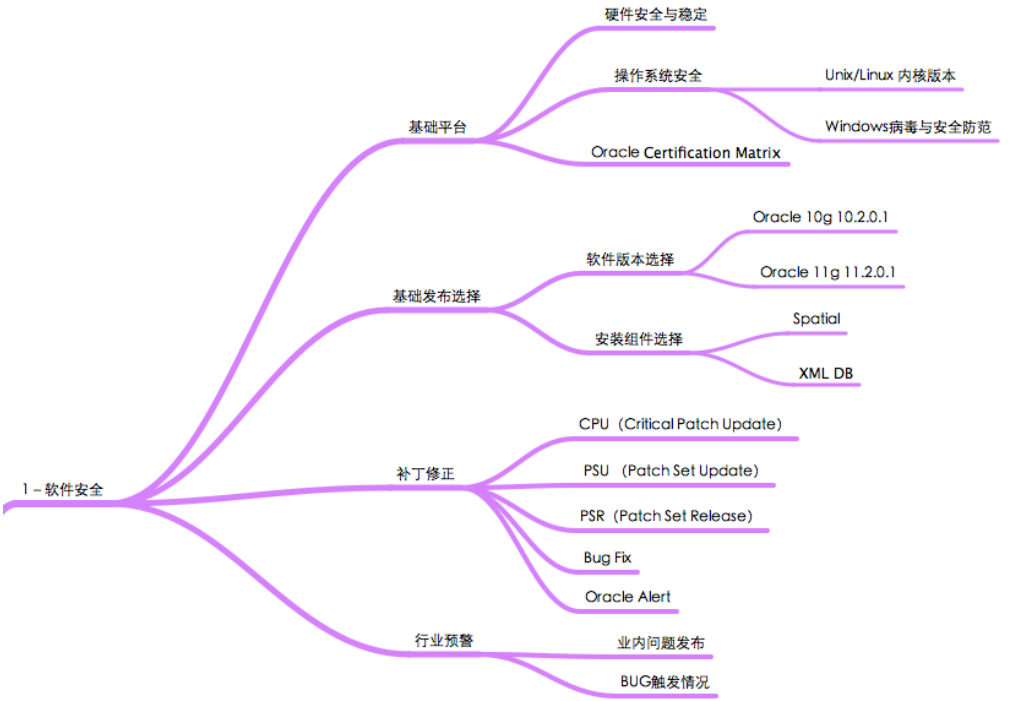
合抱之木，起于毫末

合抱之木，生于毫末；

九层之台，起于垒土；

千里之行，始于足下。

——《老子》第六十四章



“合抱之木，起于毫末”，这句话是说即便是参天大树，也是从幼苗开始生长，如果不能在事物的开端做好准备、规划，则后期的成长就可能会出现种种问题。

对于数据库来说，能否做好初始化部署与安装，及时修正已知的高风险漏洞，打好扎实的数据基础，消除数据库成长的隐患，是我们应该认真考量的重要方向之一——软件安全。

很多企业在部署软件之初，就选择了错误的版本。比如在已经发布了 Oracle Database 11.2.0.3 版本的今天，很多企业仍然在安装使用 11.2.0.1 版本，这就是非常不恰当的部署和应用。Oracle 在每一个主要补丁集中都修正了成百上千的 Bug，置如此多的问题于不顾，数据库中可能存在的种种风险由此可以想象。此外，Oracle 数据库中包含了很多产品组件，如果不需要，通常建议不要安装，因为这些组件可能会带来很多安全风险。

篇首图列举了软件安全层面可能涉及的诸多因素。