Ghost Data & Spectral Data - When is Encrypted Data Not Encrypted? And when is your data in places you didn’t expect

Jonathan Intner
J. S. Intner, Consulting, LLC
16 September, 2009
Agenda

- Introduction
- Background
  - Why encrypt?
  - Definitions
  - Problem Statement
- Test Cases:
  - Conceptually
  - 11g: tablespace
  - 10g: column
Introduction

- DBA for more than 20 years
- Certifiable:
  - OCP
  - Computer Information Systems Security Professional (CISSP)
- Currently, Project Manager and Database Architect on a Global Oracle Upgrade Project
Why Encrypt?

✦ Data Breach Regulations:
  ✦ GLBA, SOX, Ca SB1386, HIPAA, PCI-DSS, EC 45/2001 & EC Decision No 1247/2002/EC, etc.
  ✦ These breaches do occur:
    ✦ See http://www.privacyrights.org/ar/ChronDataBreaches.htm
  ✦ And they have a real cost:
    ✦ Loss of reputation
    ✦ Increased monitoring requirements and regulatory penalties
    ✦ Outright cost of $90 to $305 per record
Definitions (1)

✦ Encryption
✦ Wallet
✦ Oracle’s Advanced Security Option (ASO)
  ✦ Transparent Data Encryption (TDE)
Definitions (2)

✦ Ghost Data:

✦ Spectral Data
   ✦ Spectral data is a variant on Ghost Data, except data from one tablespace can end up in another!
   ✦ More available at: [Link](http://www.freelists.org/post/oracle-l/Ghost-Data)
Ghost Data Diagram

TableSpace 1

<table>
<thead>
<tr>
<th>Block 1</th>
<th>Block 2</th>
<th>Block 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Table 1</td>
<td>Table 1</td>
</tr>
</tbody>
</table>
Spectral Data Diagram

Block 1
Table 1
(Encrypted)

Block 2
Table 1
(Encrypted)

Block 3
Table 1
(Encrypted)

TableSpace 1
(Encrypted with TDE)

Buffer Cache

Block 1
Table 2

TableSpace 2

NYOUG: 16 September, 2009
Jonathan Intner
Spectral Data Diagram

- **Block 1**
  - **Table 1** (Encrypted)

- **Block 2**
  - **Table 1** (Encrypted)

- **Block 3**
  - **Table 1** (Encrypted)

- **TableSpace 1** (Encrypted with TDE)

- **Block 2**
  - **Table 1** (Decrypted)

- **Buffer Cache**

- **Block 1**
  - **Table 2**

- **TableSpace 2**
Spectral Data Diagram

User inserts a row into T2

Block 1
Table 1 (Encrypted)

Block 2
Table 1 (Encrypted)

Block 3
Table 1 (Encrypted)

TableSpace 1 (Encrypted with TDE)

Buffer Cache

Block 1
Table 2

TableSpace 2
Spectral Data Diagram

Block 1
Table 1
(Encrypted)

Block 2
Table 1
(Encrypted)

Block 3
Table 1
(Encrypted)

TableSpace 1
(Encrypted with TDE)

Buffer Cache

Block 1
Table 2

Block 2
Table 1
(Decrypted)

Block 2
Table 1
(Decrypted)

R T O W

R T O W

TableSpace 2

NYOUG: 16 September, 2009
Jonathan Intner
Test Case: Conceptual

- Install Oracle:
  - 11.1.0.7 with the July 2009 Critical Patch Update
  - 10.2.0.4 with the July 2009 Patch Set Update
- Shrink the buffer cache, so a single table can fill its blocks.
- Create encrypted data.
- Fully populate the buffer cache with this encrypted data.
- Insert a row into an unencrypted data and, if you used 11g’s tablespace-encryption that is available with TDE, you’ll see your encrypted data, decrypted.

Different behavior with column-level TDE data caused me to realize there’s a corollary: EVERY TIME A ROW IS INSERTED FROM A PARTIALLY-FULL BLOCK IN THE BUFFER CACHE, YOU END UP WITH “SPECTRAL” DATA IN A TABLESPACE!
Shrink the SGA & Buffer Cache.

Enable TDE:

Be careful where your wallet ends up.

The command is:

```
ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY password;
```

Use Oracle Wallet Manager (run by typing “owm” on *nix) to manage the wallet.
Configure SQLNET.ORA for the location for the Wallet.

Command to add:

```
encryption_wallet_location =
  (source =
    (method = file
      (method_data =
        (directory = <your choice> ))))
```

With 11g, it should be possible to use other locations, like an external device.
Wallet opened after DB re-started:

ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY "password";

Create an encrypted tablespace:

create tablespace fred
datafile '/u01/app/oracle/oradata/phmupg2/fred.dbf'
size 100m encryption using '3des168'
default storage (encrypt);
Used the following commands to load data into a table in that tablespace:

- create table foo tablespace fred as select * from dba_objects;
- -- execute the following until the tablespace fills:
- insert into foo (select * from foo);
Tablespaces

OS Command
Tablespaces

```
[oracle@localhost l1gtest]$ strings /u01/oradata/jsi11g/nocrypt.dbf
}||z
JSI11G
NOCRYPT
now is the time
[oracle@localhost l1gtest]$ 
```

```
[oracle@localhost l1gtest]$ strings /u01/oradata/jsi11g/crypt.dbf | more
}||z
JSI11G
CRYPT
cP)
`a)8I<
2Z<0
JLai
6.VB
 economics
```
Tablespaces

```
[oracle@localhost llgtest]$ strings /u01/oradata/jsillg/nocrypt.dbf
}{{z
JSII1G
NOCRYPT
now is the time
[oracle@localhost llgtest]$ 
```

```
[oracle@localhost llgtest]$ strings /u01/oradata/jsillg/crypt.dbf | more
}{{z
JSII1G
CRYPT
cF)
`a)8I<
2Z<0
JLai
6.VB
-251
```
Tablespaces

```
[oracle@localhost ~]$ strings /u01/oradata/jsi11g/nocrypt.dbf
}||{z
JSI11G
NOCRYPT
now is the time
[oracle@localhost ~]$ 
```

Plain Text Data

```
[oracle@localhost ~]$ strings /u01/oradata/jsi11g/crypt.dbf | more
}||{z
JSI11G
CRYPT
cF)
`a)8I<
2Z<0
JLai
6.VB
--
```

Encrypted Data
Create the problem

- Fully populate the buffer cache with data from the encrypted tablespace.

- Create a shell script that contains repeated executions of the following:
  sqlplus /nolog << EOF
  connect / as sysdba
  select * from foo;
  exit
  EOF

- While the above shell script is running, insert a row into the table in the unencrypted tablespace and examine the datafile for that tablespace with the Unix “strings” command. This confirms that unencrypted data was written to disk.
De-crypted, Encrypted data in decrypted tablespace

Searching result set

Decrypted Tablespace

Same Content

NYOUG: 16 September, 2009
Column Encryption

[oracle@phmupg t10204]$ strings nocrypt.dbf|more
}1{z
T10204
NOCRYPT
-`b_
5~,B
86M!
LqGV
!Z>H
w1d)
.qhh"
\aHr

5b5`
F"ze>m
E{m[
,MRb
now is the time
4kig
21z>
F#qY
\v8z

Tablespace that contains the table that has decrypted columns, now contains the **encrypted** data!

Here is the actual table’s data
Closing

- I understand that Oracle does consider this behavior a bug and it will be fixed in an upcoming CPU.
Cost of a data breach:

According to a 2007 study by Forrester Research. Forrester, which notes that estimating the cost of breaches is an inexact science, based its figures on a survey of 28 companies who had some sort of data breach. The costs included legal fees, call center costs, lost employee productivity, regulatory fines, loss of investor confidence and customer losses to estimate these figures. This information and the previous 3 bullets were retrieved from http://www.sans.org/reading_room/analysts_program/DLL_April08.pdf on April 22, 2009.

List of data breaches, including the one referenced on slide 4, http://www.privacyrights.org/ar/ChronDataBreaches.htm