DUDE
Where’s my data?

Presented by Jonah H. Harris
Software & Content by Kurt Van Meerbeeck
www.ora600.be
www.miracleas.dk
www.optimaldba.com
www.nrgconsulting.co.za
www hbtec.com.br
`whoami`

**Jonah H. Harris**
jonah.harris@gmail.com  
http://www.oracle-internals.com/

- *Long-time Oracle User, Developer, and Administrator*
- *Sr. Software Architect @ EnterpriseDB*
Agenda

• Data unloaders
  – what?
  – how?
    • block internals
    • unloader internals
  – demo

• Where’s my data?
  – case studies
  – unable to restore/recover
What are data unloaders?

- What are you talking about ...
  - unload data?
  - not in the sense of ETL

- Imagine your production DB crashed
  - unrecoverable
  - corrupt
  - inconsistent datafiles
  - loss of system tablespace
  - and your backup scripts weren’t as cool as you thought they were

You’ve tried everything ...
Database can’t be opened ...
WHAT DO YOU DO ?!?
What are data unloaders?

- Panic
- Cry
- Take up smoking again ...
- Call the wife – it’ll be long night ... Again ...
- Oracle support
  - spend the next 30min trying to open a severity 1 SR
  - *call* them
What are data unloaders?

- Depending on support contract / country
  1. You’re screwed
  2. You’re screwed – but maybe we can help you
     - Sending an engineer on-site
     - Take what’s left from your DB to a support office
     - Both time consuming operations -> costly
What are data unloaders?

- So what can support do what you can’t?
  - Hey – I’ve taken the Backup&Recovery course... Did I miss something?
  - DUL (Data UnLoader)
    - the myth, the legend, the Holy Grail of Oracle data recovery
    - Extracts data without the instance being up
    - written by Bernard van Dujnen, Oracle, The Netherlands (1994)
    - not a public tool -> Oracle support/consultancy
    - [www.petefinnigan.com](http://www.petefinnigan.com) : links to DUL’s user & config guide
    - (some myths are true)
What are data unloaders?

- So how do they work?
- parser/generator
  - parse Oracle blocks
  - generate flatfiles (or DMP files)
What are data unloaders?

• 3th party unloaders
  - there are only a handful
    • mydul, recovery for oracle, officerecovery
  - one of them is DUDE (www.ora600.be / www.miracleas.dk)

• Database Unloading by Data Extraction (DUDE)
  - started as open source project jDUL in 2000
  - java based – jdk/jre 1.2.x
  - Oracle versions - 7 and above
  - Since 2005 by Miracle AS, Denmark
  - Since 2007 by OptimalDBA.com (US), HBTec (Brazil), NRG Consulting (South Africa)
  - DIY (sensitive data)
What are data unloaders?

• Data unloaders - dangerous tools?
  - block level extraction – no instance needed
  - well – it’s not like BBED (block browser/block editor)
  - it can’t make things any worse
  - a hacking tool in the wrong hands?
  - file level access to datafiles necessary – strings?

  - DUDE only runs
    • on machines/database that have been probed
    • time limited
    • java code – obfuscated + encrypted classfiles
What are data unloaders?

- DUDE can only run on a specified host and database
- DUDE uses code obfuscation and class encryption
- A probe file needs to be created using DUDE_PROBE.jar
- Example dude_probe.cfg

```java
USER = "Doug Burns"
EMAIL = "dougburns@yahoo.com"
COMPANY = "Independent"
TEL = "+44xxxxxxxx"
FAX = "+44xxxxxxx"
OUTPUT_DIR = "/home/oracle/probe"

TABLESPACE "SYSTEM"
{
   DATAFILE="/usr/lib/oracle/xe/oradata/XE/system.dbf"
}

TABLESPACE "USERS"
{
   DATAFILE="/usr/lib/oracle/xe/oradata/XE/users.dbf"
}
```
Agenda

• Data unloaders
  – what?
  – how?
    • microscopic aspects - block internals
    • macroscopic aspects - unloader internals
  – demo

• Where’s my data?
  – case studies
  – unable to restore/recover
IOT’s, compressed blocks
http://www.juliandyke.com

- Blocktype (1byte)
  static final byte KTU_UNDO_HEADER = 0x01 ;
  ...
  static final byte DATA_KTB_MANAGED = 0x06 ;
  ...
  static final byte LOB_BLOCK = 0x1B ; // chunk/lobseg in non-ASSM
  static final byte PAGETABLE_MANAGED_LOB_BLOCK = 0x28 ; // chunk/lobseg in ASSM

- Data/physical objectid (4bytes)

- rowdir (2 bytes/entry)

- Others
  - # rows
  - first free row index entry
  - # tables
  - table directory

- Depends on
  - Platform
  - tablespace (ASSM, BIGFILE)

- Offset to rows

- Offset to rows

- Offset to rows

- Offset to rows
**Micro-scopic aspects – row internals**

Or what do I really, really need to make this work!

Rowheader for *non-chained/non-migrated* row in normal *heap* table

<table>
<thead>
<tr>
<th>Flag</th>
<th>Lock</th>
<th># column Pieces</th>
<th>Column header</th>
<th>Column data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>1 byte</td>
<td>1 byte</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- # col pieces = # cols if row not chained (ignore trailing nulls)
- Refers to ITL slots (interested transaction list)
- Read consistency – undo blocks are not read to rebuild the block

```java
static final int ROW_CLUSTER_KEY = 0x80;
static final int ROW_CTABLE_NUMBER = 0x40; // cluster table memer
static final int ROW_HEAD_PIECE = 0x20; // migrated row is 0x20 -> links to 0x0C (row_first_piece + row_last_piece)
static final int ROW_DELETED_ROW = 0x10;
static final int ROW_FIRST_PIECE = 0x08; // 0x28 row_head_piece + row_first_piece (start chained)
static final int ROW_LAST_PIECE = 0x04; // 0x06 last chained
static final int ROW_FROM_PREVIOUS = 0x02; // 0x03 chained
static final int ROW_CONTINUE_NEXT = 0x01; // 0x01 chained

static final int ROW_SINGLE = ROW_HEAD_PIECE + ROW_FIRST_PIECE + ROW_LAST_PIECE;
static final int ROW_MIGRATED = ROW_FIRST_PIECE + ROW_LAST_PIECE;
static final int ROW_SINGLE_DEL = ROW_HEAD_PIECE + ROW_FIRST_PIECE + ROW_LAST_PIECE + ROW_DELETED_ROW;
```
• Micro-scopic aspects – row internals

Or what do I really, really need to make this work!

Rowheader for *chained* row in normal *heap* table

<table>
<thead>
<tr>
<th>Flag</th>
<th>Lock</th>
<th># column Pieces</th>
<th>RDBA</th>
<th>Rowslot</th>
<th>Column header</th>
<th>Column data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>1 byte</td>
<td>1 byte</td>
<td>4 bytes</td>
<td>2 bytes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rowdir index of next block

Address of block containing next rowpiece

# col pieces < # cols (ignore trailing nulls)
Max 0xFF or 255 -> intra-block chaining
• Micro-scopic aspects – row internals

Or what do I really, really need to make this work!

Column header & data

- Describes size of column data
  - 1 byte
  - large columns 3 bytes including 1 marker 0xFE
  - 0xFF = NULL, 0xFE is marker

NUMBER Note:1031902.6 How does Oracle store internal numeric data?
DATE Note:69028.1 How does Oracle store the DATE datatype internally?
TIMEZONE ~ DATE
CHAR, VARCHAR, LONG – easy, but characterset conversions!
BINARY FLOAT, DOUBLE – 4/8 bytes
LOB – Tanel Põder (http://integrid.info/Poder_LOB_Internals.pdf)

Trailing NULLs are not stored
• Micro-scopic aspects – row internals

Or what do I really, really need to make this work!

Column header & data

<table>
<thead>
<tr>
<th>Column header</th>
<th>Column data</th>
<th>Column header</th>
<th>Column data</th>
</tr>
</thead>
</table>

The physical column order may not match the logical column order!!!
Data unloaders – challenges

- (Relative) database block address (R)DBA
  - 4 bytes
  - 10 bits for the (relative) file# - $2^{10} - 1 = 1023$
    - in oracle7 some wrapping applies for backwards compatibility with Oracle6
    - usually 8/2 split wrapping
  - 22 bits for the offset within the file – $2^{22} = 4.194.304$

- BIGFILE tablespace
  - one and only one datafile
  - $2^{32} = 4.294.967.296$

- Endian
  - big endian : big units first, MSB first
    - ibm powerpc, sun sparc, hp pa risc
  - little endian : little units first, LSB first
    - intel x86, alpha
- byte swapping
Data unloaders – challenges

• Micro-scopic aspects – block internals
  
  – decode block metadata
    • row directory
    • table directory (clustered table)
    • #rows, #tables
    • offsets to the metadata
  
  – decode rowheaders
    • how many columns
    • deleted row, chained row, migrated row?
  
  – decode datatypes – number, date, varchars, lobs, binary float/double

  – it’s all documented!
Data unloaders – challenges

• Macro-scopic aspects – unloader internals
  
  – meta data

  – create your own dictionary:
    • object ids, table & column names, datatypes
    • unload: obj$, tab$, col$, user$, part$, lob$

  – extentmaps: blockmaps

  – flatfiles, dump files
    • NLS issues
Data unloaders – challenges

• Macro-scopic aspects – bootstrapping

  – find base dictionary tables (obj$, tab$, user$ …)

  – find the objectid – sql.bsq

  – “MIG” utility: Oracle 7 to 8
    • migrate.bsq
    • all base dictionary tables are recreated
    • use bootstrap$
Macro-scopic aspects – dude internals

if SYSTEM is available

Commands and parameters

```
create blockmap for tablespace SYSTEM;
dump dictionary;
```

```
create blockmap for tablespace TOOLS;
```

```
create dictionary;
```

```
DICTIONARY_DIR
PARTITIONING="false"
ENABLE_LOBS="true"
```

```
Dump objectid 1450;
Dump PERFSTAT STATS$SNAPSHOT;
Dump tablespace TOOLS
```

```
USER_TABLENAME_OBJID.dmp
USER_TABLENAME_OBJID.dat
USER_TABLENAME_OBJID.ctl
```

```
DUMP_DIR
"""
```

```
SQLLDR_DIR
DDL_DIR
```

NYOUG March General Meeting 2008
Data unloaders – challenges

• Macro-scopic aspects – unloader internals

  – create your own dictionary:
    • object ids, table & column names, datatypes
    • bootstrapping: obj$, tab$, col$, user$, part$, lob$

  – what if you’ve lost system tablespace?
    • no table/column names – no datatypes!
    • heuristic scanning – data sampling
    • and a lot of guessing!
- **Macro-scopic aspects – dude internals**

if **SYSTEM** is **NOT** available

Create blockmap for tablespace **TOOLS**;

**1400**  **1444**  **1450**

3029  45003  55042

**Scan objectid 1450**
**Scan tablespace **TOOLS**;

**USER_TABLENAME_OBJID.dde**

**DICTIONARY_DIR**

**DUMP_DIR**

**DDL_DIR**

**SCANNED_OBJID.dmp**

**SCANNED_OBJID.dat**

**SCANNED_OBJID.crl**

**SCANNED_OBJID.ddl**

**DDL_DIR**

**SQLLDR_DIR**

Check data and adjust if necessary

dump SCANNED OBJECTID 16802 (COL0 NUMBER, COL1 CHAR, COL2 CHAR, COL3 DATE);
dump SCANNED OBJECTID 16802 (COL0 NUMBER, COL1 CHAR, COL2 NUMBER, COL3 DATE);
Conclusion - unloaders

• a last resort to recover your data
  - missing archivelogs
  - corruption of data dictionary or bootstrap objects
  - orphaned datafiles (or loss of system tablespace)
  - dropped tablespaces
  - truncated/dropped tables
  - dropped columns

• no guarantee
  - there’s a reason why your DB doesn’t open
  - read consistency
  - zero’d out blocks
Agenda

• Data unloaders
  – what ?
  – how ?
    • microscopic aspects - block internals
    • macroscopic aspects - unloader internals
  – demo

• Where’s my data ?
  – case studies
Where’s my data?

- Present some cases where backup/restore/recovery went terribly wrong
- Reap the harvest (recovery) you have sown (backup)

- Looking back – always a combination
  - if the seed isn’t right – you won’t harvest
    - Low budget – no money to replace hardware, use RMAN, ...
    - Human error – old school backup scripts, ignorance, lack of knowledge, time pressure, bad communication, tape circulation
    - Combination of the above
  - trigger
    - Media failure and aging hardware – disks/controller failure
    - Software failure - bugs
    - Human error – rm –fr *

- Learn from it
Case #1: “the idiot DBA”

“If there’s no activity during the night, I CAN make hot backups of the database without putting the DB in backup mode” – 1998 – the idiot DBA

- backup (seed)
  - whenever they had some spare time – no rman
  - file backup to tape without putting tablespace in backup mode

- trigger
  - media failure – lost redologs and controlfiles

- There’s no argument against stupidity

- Recovered data?
  - took datafiles to Oracle support
  - extracted data

- Red flag:
  - lack of knowledge (‘didn’t know we had to backup system tablespace’)

NYOUG March General Meeting 2008
Case #2: “Just give it a bang”

- backup (seed) & trigger
  - failing backups + media failure

- site had above average old hardware
  - disk heads sticked to the platter after cool down
  - at boot time – hit them with a hammer !!!

- surprise
  - disk crash

- Recovered data? How lucky can you be...
  - replace electronics interface of disk -> it spins up
  - got an expert on jfs/lvm on-site (www.compunix.com) -> extract datafile
  - extract Oracle data from datafile -> got it !!!

- Red flag: a hammer attached to the storage device is usually a bad sign
  Scotland DBF 2007
Case #3: “that user is not in the specs”

- backup (seed)
  - developers wrote backup script = user per user export (hardcoded)

- trigger
  - a user was added later
  - media failure

- Recovered data?
  - orphaned datafile

- Red flag: developers are in charge of backup
Case #4: “I’ll change that tape when I find some time”

- **seed**
  - non-technical people in charge of tape management/circulation
    - insert tape in drive
    - tape rewinds (thinks this is actually the backup)
    - eject
  - unmotivated people
    - leave tape in streamer for multiple days
  - doesn’t matter how good your backup mechanism is...

- **trigger**
  - could be anything – you’re screwed anyway

- **Recovered data?**
  - tape puzzling until you have all datafiles
  - datafiles from different days
  - data unloader doesn’t care if datafiles are inconsistent
  - better than nothing in some cases

- **Red flag**: tape management by non-technical or unmotivated people
Case #5: “That’s not my responsibility”

- seed – vertical responsibilities and bad communication/documentation
  - common phenomenon in large companies
  - dba, unix admin, capacity admin, SAN manager, network boys, app admins, backup administrators
  - datafiles are juggled around from SAN to SAN like a token in a token ring network
  - poor communication – nobody knew where the database resided

- trigger
  - media failure – could be anything really

- Recovered data?
  - all necessary datafiles were still there
  - no problem extracting

- Red flag: more administrators and managers than tables in the database
Case #6: “If the database crashes, we just reload”

• Datawarehousing – staging
  – monthly flatfiles
  – weekly flatfiles
  – daily flatfiles

• Seed
  – Clearly, if you have a backup of all the flatfiles – a DB backup is redundant
    .......

• Trigger
  – current redolog group corrupted

• Time might be an issue ...

• Red flag: DWH managers that are in love with sql*loader
Case #7: “Boys don’t cry – not even when they’ve lost a SYSTEM tablespace”

- **Red flag**: system, rbs/undo, temp, user, controlfiles (3) and redologs all on internal disks (%ORACLE_HOME%/database) while the data sits on a state of the art SAN/NAS/iSCSI box.

- amazingly enough – 40 to 50% of all cases have a lost system tablespace

- use system from a clone database – otherwise …
Case #8: “version based backup software – friend or enemy”

- Example IBM Tivoli Storage Manager
  - version based
  - incremental forever
    - check if file has changed
      - if so, a new version is instantiated by backing up the file
      - if not, do not backup the file

- **Red flag**: version based backup software without RMAN can be a dangerous thing

- Seed
  - version based: each backup generates a version, 1 version = 1 day

- Trigger, what if
  - backup fails half way through -> versions are out of sync
  - do multiple backups per day -> number of days to go into past decreases
Case #8: “version based backup software – friend or enemy”

Example: 7 versions – 1 backup/day – database backup fails half way through

<table>
<thead>
<tr>
<th>Version 1</th>
<th>Monday 12/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version 2</th>
<th>Tuesday 13/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version 3</th>
<th>Wednesday 14/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version 4</th>
<th>Thursday 15/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version 5</th>
<th>Friday 16/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version 6</th>
<th>Saturday 17/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version 7</th>
<th>Sunday 18/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>SYSAUX</td>
</tr>
</tbody>
</table>
Case #10: "Never knew we had to be able to restore this scenario"

• Backup (seed)
  - custom scripts
    • put tablespaces in backup mode
    • backup filesystems (incl. Archivelogs)
    • put tablespace out of backup
    - tape does not contain archive with 'end backup' commands
    - works as long as that archive is available on disk

• Trigger
  - due to fraud investigation, needed database in state of 1.5y back
  - the backup was never meant to work this way

• Recovered data?
  - had monthly tapes
  - extracted data

• Red flag: bad luck – but at least make the tape independent of any online data
Have you ever not been able to recover ???

- YES
- zero’d out blocks
- windows crash and memory mapped files
Conclusion

- Use common sense
- Use RMAN
- Lack of knowledge – hire a consultant
- Practise restore/recovery
Questions

www.ora600.be
www.miracleas.dk
www.optimaldba.com
www.nrgconsulting.co.za
www.hbtec.com.br