Lessons Learned while Pushing the Limits of SecureFile LOBs

by Jacco H. Landlust
Jacco H. Landlust

- 36 years old
- Deventer, the Netherlands
Jacco H. Landlust / iDBA

- Degree in Business Informatics and Economics
- Architecture, Clustering, High Availability, Performance & Management
- Oracle since 2000
- Oracle ACE since 2006
- Independent Red Stack Administrator since 2010
This is not a “SecureFile LOB overview” presentation!
Agenda

- WebCenter Content / UCM
- Short introduction of SecureFile LOBs
- Test some SecureFile LOB features
- Q & A
WebCenter Content
10g configuration
10g configuration

- 35k online users
10g configuration

- 35k online users
- Concurrency issues: Lots of row lock contention
10g configuration

- 35k online users
- Concurrency issues: Lots of row lock contention

Fixed in 11g
10g configuration

- 35k online users
- Concurrency issues: Lots of row lock contention
- 40 million unique documents ≈ 24 TB

Fixed in 11g
10g configuration

- 35k online users
- Concurrency issues: Lots of row lock contention
- 40 million unique documents ~= 24 TB
- Metadata and content separated, content was stored on GPFS

Fixed in 11g
10g configuration

- 35k online users
- Concurrency issues: Lots of row lock contention
- 40 million unique documents ~ 24 TB
- Metadata and content separated, content was stored on GPFS

One database with SecureFile LOBs

Fixed in 11g
Introducing SecureFile LOBs
Introducing SecureFile LOBs

- SecureFile LOBs eliminate the distinction between structured and unstructured content storage.
Introducing SecureFile LOBs

- SecureFile LOBs eliminate the distinction between structured and unstructured content storage.
- SecureFile LOBs is a new re-architecture featuring entirely new disk formats, network protocol, space management, redo and undo formats, buffer caching, and I/O subsystem.
Introducing SecureFile LOBs

- SecureFile LOBs eliminate the distinction between structured and unstructured content storage.
- SecureFile LOBs is a new re-architecture featuring entirely new disk formats, network protocol, space management, redo and undo formats, buffer caching, and I/O subsystem.
- SecureFile LOBs delivers substantially improved performance along with optimized storage for unstructured data inside the Oracle database.
Introducing SecureFile LOBs
Introducing SecureFile LOBs

- Tablespaces must be managed by ASSM
Introducing SecureFile LOBs

- Tablespaces must be managed by ASSM
- Easier management, lesser user-tuned parameters
Introducing SecureFile LOBs

- Tablespaces must be managed by ASSM
- Easier management, lesser user-tuned parameters
- One database parameter (plus some hidden ones)
Introducing SecureFile LOBs

• Tablespaces must be managed by ASSM
• Easier management, lesser user-tuned parameters
• One database parameter (plus some hidden ones)
• Lobs from Oracle Database 10g and prior releases are still supported and will now be referred to as ‘BasicFiles’.
Introducing SecureFile LOBs

- Tablespaces must be managed by ASSM
- Easier management, lesser user-tuned parameters
- One database parameter (plus some hidden ones)
- Lobs from Oracle Database 10g and prior releases are still supported and will now be referred to as ‘BasicFiles’.
- Certain features require extra licenses (deduplication, compression, encryption)
Test setup today
Test setup today

- VirtualBox VM
Test setup today

- VirtualBox VM
- 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
Test setup today

- VirtualBox VM
  - 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
  - ASM disk is fully allocated
Test setup today

- VirtualBox VM
- 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
- ASM disk is fully allocated
Test setup today

- VirtualBox VM
  - 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
  - ASM disk is fully allocated
- Oracle Enterprise Linux 5.8

No thin provision to minimize VM overhead
Test setup today

- VirtualBox VM
  - 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
  - ASM disk is fully allocated
- Oracle Enterprise Linux 5.8
  - kernel 2.6.39-300.17.3.el5uek

No thin provision to minimize VM overhead
Test setup today

- VirtualBox VM
  - 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
  - ASM disk is fully allocated
- Oracle Enterprise Linux 5.8
  - kernel 2.6.39-300.17.3.el5uek
- Oracle RDBMS 11.2.0.2.6 with ASM external redundancy

No thin provision to minimize VM overhead
Test setup today

- VirtualBox VM
- 2 cores, 4 GB RAM, 3 virtual disks (OS, Software, ASM)
- ASM disk is fully allocated
- Oracle Enterprise Linux 5.8
  - kernel 2.6.39-300.17.3.el5uek
- Oracle RDBMS 11.2.0.2.6 with ASM external redundancy

No thin provision to minimize VM overhead

Had to upgrade to 11.2.0.3 because of bug 13775960 - "enqueue hash chains" latch contention for delete/insert Securefile workload
Database
Database

- AL32UTF8
Database

- AL32UTF8

Mandatory by Fusion Middleware installers
Database

- AL32UTF8
- MEMORY_TARGET=2G

Mandatory by Fusion Middleware installers
Database

- AL32UTF8
- MEMORY_TARGET=2G

Mandatory by Fusion Middleware installers

Automatic unless
Database

- AL32UTF8
- MEMORY_TARGET=2G
- In archivelog mode

Mandatory by Fusion Middleware installers

Automatic unless
WebCenter repository
WebCenter repository

- By default smallfile tablespaces
WebCenter repository

- By default smallfile tablespaces

Replace with bigfile tablespace
WebCenter repository

- By default smallfile tablespaces
- By default 8k blocksize

Replace with bigfile tablespace
WebCenter repository

- By default smallfile tablespaces
- By default 8k blocksize

Replace with bigfile tablespace

Choose based on content (typically 8k turns out okay)
WebCenter repository

- By default smallfile tablespaces
- By default 8k blocksize
- By default basicfile LOB

Replace with bigfile tablesapce

Choose based on content (typically 8k turns out okay)
WebCenter repository

- By default smallfile tablespaces
- By default 8k blocksize
- By default basicfile LOB
- FileStorage table created from within WebCenter Content

Replace with bigfile tablespace

Choose based on content (typically 8k turns out okay)
WebCenter repository

- By default smallfile tablespaces
- By default 8k blocksize
- By default basicfile LOB
- FileStorage table created from within WebCenter Content

- Replace with bigfile tablespace
- Choose based on content (typically 8k turns out okay)
- Replace with SecureFile LOB
## ASM: compatible.rdbms

<table>
<thead>
<tr>
<th>Redundancy</th>
<th>compatible.rdbms=10.1</th>
<th>compatible.rdbms=11.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>External</td>
<td>16 TB</td>
<td>140 PB</td>
</tr>
<tr>
<td>Normal</td>
<td>5.8 TB</td>
<td>23 PB</td>
</tr>
<tr>
<td>High</td>
<td>3.9 TB</td>
<td>15 PB</td>
</tr>
</tbody>
</table>
## ASM: compatible.rdbms

<table>
<thead>
<tr>
<th>Redundancy</th>
<th>compatible.rdbms=10.1</th>
<th>compatible.rdbms=11.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>External</td>
<td>16 TB</td>
<td>140 PB</td>
</tr>
<tr>
<td>Normal</td>
<td>5.8 TB</td>
<td>23 PB</td>
</tr>
<tr>
<td>High</td>
<td>3.9 TB</td>
<td>15 PB</td>
</tr>
</tbody>
</table>

Default value
ORA-15095: reached maximum ASM file size (16384 GB)
ORA-600: internal error code, arguments: [krccfl_bitmap_too_small], [19], [4294340465], [4], [4366], [4366], []
ORA-600: internal error code, arguments: [krccfl_bitmap_too_small], [19], [4294340465], [4], [4366], [4366], []

Only when using block change tracking
redo_log & log_buffer
redo_log & log_buffer

- Set log_buffer to maximum (256MB on 64-bit Linux) to handle peak/burst load
redo_log & log_buffer

- Set log_buffer to maximum (256MB on 64-bit Linux) to handle peak/burst load

Only penalty seems small memory overhead
redo_log & log_buffer

- Set log_buffer to maximum (256MB on 64-bit Linux) to handle peak/burst load
- Default redo_log files too small for high concurrency and lots of data loading, enlarge to at least 1GB with 3 logfiles

Only penalty seems small memory overhead
**redo_log & log_buffer**

- Set log_buffer to maximum (256MB on 64-bit Linux) to handle peak/burst load.
- Default redo_log files too small for high concurrency and lots of data loading, enlarge to at least 1GB with 3 logfiles.

1GB is arbitrary number, Monitor log file switches in alert log.

Only penalty seems small memory overhead.

dinsdag 9 april 13
Partitioning
Partitioning

- Similar to regular tables / BasicFile LOBs
Partitioning

- Similar to regular tables / BasicFile LOBs
- All LOB segment partitions must have same blocksize
Partitioning

- Similar to regular tables / BasicFile LOBs
- All LOB segment partitions must have same blocksize
Partitioning

- Similar to regular tables / BasicFile LOBs
- All LOB segment partitions must have same blocksize
- Can ease backup & recovery strategy, e.g. by interval partition
Partitioning

- Similar to regular tables / BasicFile LOBs
- All LOB segment partitions must have same blocksize
- Can ease backup & recovery strategy, e.g. by interval partition

When moving subpartition on interval partitioned table:
ORA-00600: internal error code, arguments: [kkpod nextFrag], [10], [20], [1], [1], [93891], [], [], [], [], [], [], []
Investigating SecureFile LOB features
Shared IO Pool
Shared IO Pool

• Used for large I/O operations on SecureFile Lobs
Shared IO Pool

- Used for large I/O operations on SecureFile Lobs
- Shared memory segment
Shared IO Pool

- Used for large I/O operations on SecureFile Lobs
- Shared memory segment
Shared IO Pool

- Used for large I/O operations on SecureFile Lobs
- Shared memory segment
- If Shared IO Pool is exhausted, memory is claimed from PGA
Shared IO Pool

- Used for large I/O operations on SecureFile Lobs
- Shared memory segment
- If Shared IO Pool is exhausted, memory is claimed from PGA

Automatic Shared Memory Management required

Can be monitored from v$securefile_timer
Demo: Shared IO Pool
Caching
Caching

- CACHE - LOB data is placed in the buffer cache
Caching

- CACHE - LOB data is placed in the buffer cache
- CACHE READS - LOB data in only placed in buffer cache during read operation, not write operations
Caching

- CACHE - LOB data is placed in the buffer cache
- CACHE READS - LOB data in only placed in buffer cache during read operation, not write operations
- NOCACHE - LOB data is not placed in the buffer cache
Caching

- CACHE - LOB data is placed in the buffer cache
- CACHE READS - LOB data in only placed in buffer cache during read operation, not write operations
- NOCACHE - LOB data is not placed in the buffer cache
- CACHE and NOLOGGING not supported as combination
Caching

- CACHE - LOB data is placed in the buffer cache
- CACHE READS - LOB data is only placed in buffer cache during read operation, not write operations
- NOCACHE - LOB data is not placed in the buffer cache
- CACHE and NOLOGGING not supported as combination
Caching

- CACHE - LOB data is placed in the buffer cache
- CACHE READS - LOB data in only placed in buffer cache during read operation, not write operations
- NOCACHE - LOB data is not placed in the buffer cache
- CACHE and NOLOGGING not supported as combination
- NOCACHE when lots of documents are stored but not often retrieved
Space Management
Space Management

- SMCO background process
Space Management

- SMCO background process
- Wnnn processes are SMCO slaves
Space Management

- SMCO background process
- Wnnn processes are SMCO slaves
- Tablespace-level space (extent) pre-allocation
Space Management

- SMCO background process
- Wnnn processes are SMCO slaves
- Tablespace-level space (extent) pre-allocation
- SecureFile LOB segment pre-allocation
Space Management

- SMCO background process
- Wnnn processes are SMCO slaves
- Tablespace-level space (extent) pre-allocation
- SecureFile LOB segment pre-allocation
- SecureFile LOB segment space reclamation
Space Management

- SMCO background process
- Wnnn processes are SMCO slaves
- Tablespace-level space (extent) pre-allocation
- SecureFile LOB segment pre-allocation
- SecureFile LOB segment space reclamation

Sessions don’t have to wait for space allocation / deallocation operations, because this is proactively done.
Demo: Space Management
Small extents
Small extents

- Minimal extent size is 5 blocks (8Kb blocksize = 40Kb)
Small extents

- Minimal extent size is 5 blocks (8Kb blocksize = 40Kb)
- ORA-60019: Creating initial extent of size 5 in tablespace of extent size 14
Small extents

- Minimal extent size is 5 blocks (8Kb blocksize = 40Kb)
- ORA-60019: Creating initial extent of size 5 in tablespace of extent size 14
- So minimum extent size is 14 blocks (8Kb blocksize = 112Kb)
Small extents

- Minimal extent size is 5 blocks (8Kb blocksize = 40Kb)
- ORA-60019: Creating initial extent of size 5 in tablespace of extent size 14
- So minimum extent size is 14 blocks (8Kb blocksize = 112Kb)
- ORA-00600: internal error code, arguments: [ktssladdfcb-bsz], [3], [], [], [], [], [], [], [], [], [], []
Small extents

• Minimal extent size is 5 blocks (8Kb blocksize = 40Kb)
• ORA-60019: Creating initial extent of size 5 in tablespace of extent size 14
• So minimum extent size is 14 blocks (8Kb blocksize = 112Kb)
• ORA-00600: internal error code, arguments: [ktsladdfcb-bsz], [3], [], [], [], [], [], [], [], [], [], []
• Real minimum extent size for SecureFile LOBs = (14 * 8Kb) + 1 = 112Kb + 1 = 114689
high VKTM CPU usage
high VKTM CPU usage

- Virtual keeper of time provides wall-clock time and reference time for other sessions/processes
high VKTM CPU usage

- Virtual keeper of time provides wall-clock time and reference time for other sessions/processes
- Gets system time every 10 ms
high VKTM CPU usage

- Virtual keeper of time provides wall-clock time and reference time for other sessions/processes
- Gets system time every 10 ms
- Process priority tunable by modifying _high_priority_processes parameter
high VKTM CPU usage

- Virtual keeper of time provides wall-clock time and reference time for other sessions/processes
- Gets system time every 10 ms
- Process priority tunable by modifying `_high_priority_processes` parameter
- `_high_priority_processes = [VKTM|LMS*|LGWR]`
high VKTM CPU usage

- Virtual keeper of time provides wall-clock time and reference time for other sessions/processes
- Gets system time every 10 ms
- Process priority tunable by modifying `_high_priority_processes` parameter
- `_high_priority_processes` = `[VKTM|LMS*|LGWR]`

VKTM|LMS* by default on single instance
Bunch of SR’s

SR 3-5003949261: Heavy Library cache lock contention on 11.2.0.2 RAC database

|--- SR 3-5249785361: High average times on gc waits
|--- SR 3-5312761310: enq: HW - contention excessive avg. wait time in rac4W
|--- SR 3-5255677303: Process waiting on disk file i/o operation and blocking 30 sessions

SR 3-4963615411: 11.2.0.2 RAC database: Adding disks to Diskgroup, causes enq HW:Contention on the database Inserts

|--- SR 3-5128746431: LOB insert causing high "enq: HW - contention" waits
|--- SR 3-5257318187: NAR : Child SR for RAC Performance
Suggested changes by support

1) Increase db_writer_processes from 2 to 4
2) Reduce the "db_multiblock_read_count to 8
3) Set _buffer_busy_wait_timeout=2 (= 20 ms) due to Bug 11930616 - sporadic buffer busy waits
4) Suggestion to implement jumbo frames
5) Apply patches: --
   Patch 9801919: ENQ: HW - CONTENTION WAIT TIME IS VERY LONG
   Patch 9671271 - All active instances used in calculation of dop when parallel_force_local=true / High
   version count on PX_MISMATCH
6) Bug 13698526 : 11.2.0.2 RAC DATABASE: ADDING DISKS TO DISKGROUP, CAUSES ENQ
   HW:CONTENTION --> has no update by ASM development team.
7) Tune log file sync -- probably seperate diskgroup for redo and adjust the storage FA ports to assign
   less busy ports.
8) Trying to create partition (qespcCreatePartition) which explains why we need library cache lock in
   exclusive mode. Other processes are waiting for file resize - kfnC SlaveFileResize in stack. Slave
   process spawned dynamically by SMCO [Smco (Space Management Coordinator) And Autoextend On
   Datafiles (Doc ID 743773.1)]
SMCO: pre-allocate extent

If available spaces in table is less than 5 %
SMCO: pre-allocate extent

If available spaces in table is less than 5%

Preallocate 5% space until max 90% of tablespace maxsize
SMCO: pre-allocate extent

If available spaces in table is less than 5%

Preallocate based on autoextent next size

Preallocate 5% space until max 90% of tablespace maxsize
SMCO: pre-allocate extent

If available spaces in table is less than 5% → Preallocate 5% space until max 90% of tablespace maxsize

Preallocate based on autoextent next size → example: 50M preallocation = ceil(50M / 1M) = 50 operations
SMCO: pre-allocate extent

If available spaces in table is less than 5%

Preallocate based on autoextent next size

Preallocate 5% space until max 90% of tablespace maxsize

example: 50M preallocation = ceil(50M / 1M) = 50 operations

what if datafile is 10TB?
SMCO: pre-allocate extent

If available spaces in table is less than 5%  
Preallocate based on autoextent next size  
what if datafile is 10TB?

Preallocate 5% space until max 90% of tablespace maxsize

example: 50M preallocation = ceil(50M / 1M) = 50 operations

10TB * 5% = 500GB  
ceil(500G / 1M) = 512000 operations
SMCO: pre-allocate extent

What if my users insert would trigger an extent creation and SMCO starts pre-allocating?
SMCO: pre-allocate extent

What if my users insert would trigger an extent creation and SMCO starts pre-allocating?

enq: TX contention until SMCO is finished pre-allocating
SMCO: pre-allocate extent

What if my users insert would trigger an extent creation and SMCO starts pre-allocating?

enq: TX contention until SMCO is finished pre-allocating

when using partitioning: library cache lock
SMCO: pre-allocate extent

- AUTOEXTEND Grows To Full Size Without Reason [ID 1459097.1]
- Wnnn processes consuming high CPU [ID 1492880.1]
- Bug 11710238 - Instance crash due to ORA-600 [1433] for SMCO messages [ID 11710238.8]
- SMCO (Space Management Coordinator) For Autoextend On Datafiles And How To Disable/Enable [ID 743773.1]
Solution?
Solution?

- Configure `_enable_space_prelallocation`, but has unintended side effects
Solution?

- Configure _enable_space_preallaction, but has unintended side effects
- Pre-allocate space manually so SMCO doesn’t kick in
Solution?

- Configure `_enable_space_preallaction`, but has unintended side effects
- Pre-allocate space manually so SMCO doesn’t kick in

Can be automated
Filesystem_like_logging
Filesystem_like_logging

- Replaces nologging for SecureFile LOBs
Filesystem_like_logging

• Replaces nologging for SecureFile LOBs

Even with nologging operation some redo is created
Filesystem_like_logging

- Replaces nologging for SecureFile LOBs
- SecureFile LOBs only logs metadata similar to metadata journaling of file systems

Even with nologging operation some redo is created
Filesystem_like_logging

- Replaces nologging for SecureFile LOBs
- SecureFile LOBs only logs metadata similar to metadata journaling of file systems
- Ensures that data is recoverable after server failure
Filesystem_like_logging

- Replaces nologging for SecureFile LOBs
- SecureFile LOBs only logs metadata similar to metadata journaling of file systems
- Ensures that data is recoverable after server failure
**Filesystem_like_logging**

- Replaces nologging for SecureFile LOBs
- SecureFile LOBs only logs metadata similar to metadata journaling of file systems
- Ensures that data is recoverable after server failure
- force logging overrides filesystem_like_logging

Even with nologging operation some redo is created

By reading SecureFile LOB index
Demo: filesystem_like_logging
Block types for SecureFiles

1. NGLOB: Lob Extent Header
2. NGLOB: Segment Header
   - Second block of the first extent
   - Highwater Mark, Extent Map, Administration of Hash Bucket Blocks
3. NGLOB: Extent Map
4. NGLOB: Committed Free Space
5. NGLOB: Persistent Undo
6. NGLOB: Hash Buckets – variable chunk-size
   - 7 Buckets for chunks of different sizes: 2k to 32K, 32k to 64k, 64k to 128k, 128k to 256k, 256k to 512k, 512k to 1m, 1m to 64m
Compression
Compression

- SecureFile compression ≠ table compression
Compression

- SecureFile compression != table compression
- Oracle automatically detects if data is compressible
Compression

- SecureFile compression != table compression
- Oracle automatically detects if data is compressible
- NOCOMPRESS | COMPRESS MEDIUM | COMPRESS HIGH
Compression

- SecureFile compression != table compression
- Oracle automatically detects if data is compressible
- `NOCOMPRESS | COMPRESS MEDIUM | COMPRESS HIGH`
- For partitioned tables, compression occurs at partition level
Compression

• SecureFile compression != table compression
• Oracle automatically detects if data is compressible
• NOCOMPRESS | COMPRESS MEDIUM | COMPRESS HIGH
• For partitioned tables, compression occurs at partition level

Can configure compression level per partition
Compression

- SecureFile compression != table compression
- Oracle automatically detects if data is compressible
- NOCOMPRESS | COMPRESS MEDIUM | COMPRESS HIGH
- For partitioned tables, compression occurs at partition level
- Cost versus benefit

Can configure compression level per partition
Demo: Compression
Deduplication
Deduplication

- Eliminate multiple redundant copies of the same data
Deduplication

- Eliminate multiple redundant copies of the same data
- Duplicate detection does not span across partitions or subpartitions
Deduplication

- Eliminate multiple redundant copies of the same data
- Duplicate detection does not span across partitions or subpartitions
- Oracle uses a secure hash index to detect duplicate SecureFile data
Deduplication

- Eliminate multiple redundant copies of the same data
- Duplicate detection does not span across partitions or subpartitions
- Oracle uses a secure hash index to detect duplicate SecureFile data

Stored in same tablespace as the LOB segments
Demo: Deduplication

8_deduplication_rate.sql
9_deduplication_cost.sql
rerun 9 with complete oracle docs to show that more files means slower dedup
Summary
Summary

- Setup your database with care
Summary

- Setup your database with care

ASM, redo logs, log_buffer, db_securefile
Summary

- Setup your database with care
- Test and analyze licensable features carefully
Summary

- Setup your database with care
- Test and analyze licensable features carefully

ASM, redo logs, log_buffer, db_securefile

Compression & deduplication are not always useful
Summary

- Setup your database with care
- Test and analyze licensable features carefully
- Develop a sizing strategy & preallocate space yourself

ASM, redo logs, log_buffer, db_securefile

compression & deduplication are no always usefull
Summary

- Setup your database with care
- Test and analyze licensable features carefully
- Develop a sizing strategy & preallocate space yourself

ASM, redo logs, log_buffer, db_securefile

Compression & deduplication are not always useful

Block size, SMCO pre-allocation
Summary

- Setup your database with care
- Test and analyze licensable features carefully
- Develop a sizing strategy & preallocate space yourself
- Monitor your production environment carefully

Additional Tips:

- ASM, redo logs, log_buffer, db_securefile
- Compression & deduplication are not always usefull
- Block size, SMCO pre-allocation