Breaking Oracle

Simulating failures for testing and diagnostic practice

Jeremiah Wilton

ORA-600 Consulting

About Jeremiah

- Amazon's first DBA 1997-2004
- Working with Oracle since 1994
- Owner, ORA-600 Consulting http://www.ora-600.net
  - Architecture, scaling, performance
  - Availability, stability, complex recovery
  - Training, seminars, recruiting
- UW Certificate Program instructor
- Internals and nontrivial issue resolution

OakTable.net

ORACLE Certified Master

ORA-600 CONSULTING

Problem profiles

- Hangs
  - Single-sesson
  - Multi-session
  - Whole instance
  - Multi-instance
- Spins
  - Server process
  - Background process
- Crashes
  - Session/server/process
  - Whole instance
  - ORA-600, ORA-7445
- Curruption/data loss
  - Files
  - Blocks
  - Logical
  - Diabolical

Rationale

- Substitute for real-world ordeals
- Hard to find good troubleshooters
- High cost of outages
- Opportunity for improvement
- Obscurity of diagnostic skills
  - Not a standard DBA skill
  - Not well documented
- Inadequacy of OWS first-line
- Fun, exciting

Inducing Load

- Need a realistic load to induce hangs, etc.
- Resource contention is a problem of concurrency
- Under load, problems get worse
- Helps find scaling limits of a system
- An inactive site is no excuse for not learning
- Many recent options available

Induced Load: Options

- Generated workload
  - Can be turned up to exhaust server resources
- Recorded workload
  - Your application's true load
  - Less opportunity to ratchet up
- Application service loaders
  - HP LoadRunner, OpenSTA
- Database-only loaders
  - Database Replay, HammerOra, Swingbench

Swingbench

- Open-source tool by Dominic Giles (Oracle UK)
- Synthetic load harness
- Useful canned workloads
  - Order Entry
  - Calling Circle
- Possible to roll your own workload
- Quick and easy to set up
- http://www.dominicgiles.com/swingbench

Database Replay

- Part of 11*g* Real Application Testing
- Capture from earlier versions
  - 9.2.0.8, 10.2.0.3, 10.2.0.4
- Allows workload to resemble real application
- Allows subsetting by user, app, etc.
- Premium option
- **Primarily for change assurance**

Hangs

- One or more sessions getting "stuck"
- Really means waiting on something
- Locks, latches, I/O, object serialization
- Hanging sessions may be holding resources needed by others
- Work ethic of waits
- Long (legitimate) waits vs. hangs
  - Oracle's view
  - Customer's view

Whole-instance hang

- Hang I/O calls by processes that can't time out

```
root@dbhost# mount -F nfs -o rw \
           localhost:/opt/oracle/oradata/od08/bct /mnt/orabct

SYS> alter database enable block change tracking
           using file '/mnt/orabct/bct.ora';

user@dbclient$ ./charbench

root@dbhost# /etc/init.d nfs.server stop

SYS> column program format a15 trunc
SYS> column event format a45
SYS> select sid, program, event, state,
            seconds_in_wait, blocking_session
            from v$session where type != 'BACKGROUND'
```

- CTWR holds resources needed by running sessions

Spins

- Endless loops
- Process may be hanging or not
- Found with top or ps
- Consumes CPU resources
- If hanging may be holding resources needed by others

Server process spins

- Hang and spin in regular expression search

```
SQL> select 1 from dual where regexp_like(' ','^*[ ]*a');

oracle@dbhost$ ps -eo pid,pcpu,args | sort -n +1 | tail -10

SQL> @waits
```

Background process spins

- Spinning background procs can't always be killed without terminating the instance

```
oracle@db02$ ps -eo pid,s,args | grep ora_arc
oracle@db02$ kill -STOP `ps -eo pid,args | grep ora_arc \
             | grep -v grep | awk '{print $1}'`
oracle@db02$ ps -eo pid,s,args | grep ora_arc
SQL> select group#, sequence#, archived, status from v$log
     order by sequence#;
SQL> alter system switch logfile;
SQL> alter system switch logfile;
SQL> alter system switch logfile;
oracle@db02 $ ps -eo pid,pcpu,args | sort -n +1 | tail -10
SQL> column event format a45
SQL> select event, state, seconds_in_wait from v$session
     where type = 'BACKGROUND' and program like '%LGWR%';
```

11*g* Background Processes: Which ones crash the instance?

Process Name	Description
ACMS	Atomic controlfile to memory server
ARC*n*	Redo log archivers
CJQ*n*	Job scheduler coordinator
CKPT	Checkpoint
D*nnn*	Dispatchers
DBRM	Resource manager process
DBW*n*	Database writer processes
DIA0	Diagnosibility process 0
DIAG	Diagnosibility coordinator
FDBA	Flashback data archiver process
Jnnn	Job scheduler processes
LGWR	Redo log writer
LMD*n*	Global enqueue service daemons
LMON	Global enqueue service monitor
MMAN	Memory manager

Process Name	Description
MMNL	Manageability Monitor Process 2
MMON	Manageability Monitor Process
PING	Interconnect latency measurement
PMON	Process monitor
PSP*n*	Process spawners
Q*nnn*	Queue cleanup processes
QMNC	Queue coordinator
RECO	Distributed recovery process
RMS*n*	RAC management server
RVWR	Recovery writer
S*nnn*	Shared servers
SMCO	Space management coordinator
SMON	System monitor process
VKTM	Virtual keeper of time process
W*nnn*	Space management processes

ORA-600 CONSULTING

Crashes

- Usually ORA-00600 and ORA-07445
- Single process crash *can* take down whole instance
- ORA-00600: internal error code, arguments: [] [] [] []
  - First argument tells you calling function or numeric identifier
  - Additional arguments provide more information
  - Process/session does not always die
  - Not necessarily an emergency
- ORA-07445: exception encountered: core dump [] []
  - Core dump
  - First argument tells you where in the code (10*g*+)
  - Second argument is the signal (kill -l)
  - Additional arguments provide more information

ORA-00600 Example

- Simplest case in PL/SQL

```
SQL> declare
       a exception;
       pragma exception_init(a,-600);
     begin
       raise a;
     end;
```

- Nicer, lets you specify the arguments

```
SQL> oradebug unit_test dbke_test dde_flow_kge_ora ouch! 0 0
```

Bug that raises ORA-00600

- Bug 6073325: SELECT QUERY WITH CONNECT BY PRIOR FAILS WITH ORA-00600 [KKQCBYDRV:1]

```
SQL> select 1 from sys.table_privileges tp, user_objects uo
     where tp.grantee in
        (select 1 from sys.dba_role_privs
         connect by prior granted_role = grantee
         start with  grantee = 'scott');
```

- Raises ORA-600, but we are sill connected
- Not all -600 errors are fatal (most are not)
- Just a unhandled exception - no reason to panic

ORA-07445 Example

- Simplest case: send a signal

```
SQL> select spid from v$process p, v$session s
     where p.addr = paddr
     and sid = sys_context('USERENV','SID');
oracle@db02$ kill -SEGV 2513
```

- Use PL/SQL

```
SQL> declare
       a exception;
       pragma exception_init(a,-7445);
     begin
       raise a;
     end;
```

Real ORA-07445 bug

- Bug 6244173: ORA-07445 IN QEESTRAVERSEEXPR FOR HIERARCHICAL QUERY

```
SQL> create table t2(col1 varchar2(60));
SQL> create table t1(c1 varchar2(60),
                     c2 varchar2(1),
                     c3 varchar2(60),
                     c4 varchar2(60));
SQL> explain plan for
     select 1 from t1 a, t2 b ,t1 c
     where b.col1 = 'xxslc_department'
     and a.c1 not between c.c3 and  c.c4
     start with a.c2='p'
     connect by prior a.c1 between a.c3 and a.c4;
```

- Raises ORA-3113, so we look in alert log…
- Nature of a crashed process to generate a disconnect
- Continued use of dead connection gives app:
  – ORA-3114: Not connected to Oracle
  – ORA-1041: internal error. hostdef extension doesn't exist
    - oerr ora 1041 - Call support!

Whole-instance crashes

- Something causes a required background process to exit
- ORA-600, ORA-7445, I/O errors, etc.
  - Can actually be any error that prevents the next step
- Some will restart, some crash the instance

Instance crashes

- Simple case: kill an essential background process (tail the alert log)

```
oracle@db02$ ps -eo pid,args | grep ora_ckpt | grep -v grep
oracle@db02$ kill -KILL <pid>
```

- Simple case: send a SIGSEGV or SIGBUS to an essential background process

```
oracle@db02$ ps -eo pid,args | grep ora_dbrm | grep -v grep
oracle@db02$ kill -SEGV <pid>
```

  – Raises ORA-07445

Instance crashes

- Cause fatal errors in essential background processes

```
SQL> select pid, program, background from v$process
     where background = 1;
SQL> oradebug setorapid 16
SQL> oradebug call kgeasnmierr 4455547624 18446744071472029760
     18446744071562043788 2 1 1
```

Corruption

- Physical
  - File headers
  - Data blocks
  - Controlfiles, logfiles, other logs
  - Caused by Oracle, O/S and hardware bugs
- Logical
  - Application tables
  - Data dictionary

Data block corruption

- Simple example: garbage into a block
- Find a block in a known table

```
SQL> select min(dbms_rowid.rowid_block_number(rowid))
     from soe.customers;
SQL> select customer_id, cust_email from soe.customers
     where dbms_rowid.rowid_block_number(rowid) = 12;
oracle@db02 $ dd if=/opt/oracle/oradata/od08/soe.dbf bs=8192 iseek=12 \
              count=1 | strings | grep Sachin.Neeson@oracle.com
oracle@db02$ dd if=$ORACLE_HOME/bin/oracle  \
                of=/opt/oracle/oradata/od08/soe.dbf \
                bs=8192 oseek=12 count=1 conv=notrunc
1+0 records in
1+0 records out
SQL> alter system checkpoint;
```

- Check the alert log - no errors!
- Read the block

```
SQL> select customer_id, cust_email from soe.customers
     where dbms_rowid.rowid_block_number(rowid) = 12;
SQL> alter system flush buffer_cache;
SQL> select customer_id, cust_email from soe.customers
     where dbms_rowid.rowid_block_number(rowid) = 12;
```

- Restore data block (read again)

```
RMAN> blockrecover datafile '/opt/oracle/oradata/od08/od08/soe.dbf' block 12;
```

Other vulnerable files

- Archived redo logs
- Flashback logs
- Flashback archives
- Block change tracking file
- Backups

Logical corruption

- Erroneously changed data
  - Missing/incorrect predicate (where clause)
- Human error/application bug
- Oracle bug (wrong results)

- Many tools to resolve
  - Flashback query
  - Flashback transaction
  - Flashback table
  - Flashback database
  - Log Miner
  - Traditional point-in-time recovery
  - Mini-clone recovery

Logical corruption

- User oops: missing where clause

```
SQL> update customers set cust_first_name = 'Nimrod'
     where rownum < 1000;
SQL> commit;
SQL> select versions_startscn, versions_endscn, versions_xid
     from customers
     versions between timestamp sysdate-(.25/24) and sysdate
     where cust_first_name = 'Nimrod';
SQL> select undo_sql from flashback_transaction_query
     where xid = '00090015000003A1'
```

- Quality resolution requires examining "versions between" to get exact SCN of changes (undo_retention).
- Don't forget that there may have been subsequent changes to rows

Q&A