



Does VPD, FGA or audit really cause performance issues

Pete Finnigan, Principal Consultant

SIEMENS

Insight Consulting

Introduction

👉 My name is Pete Finnigan

👉 I specialise in researching, auditing and securing Oracle databases

👉 I am going to keep it reasonably simple and not too technical

👉 Audit is becoming a key issue for most organisations

👉 Technologies like FGA and VPD are little used due to pre-conceptions, misunderstandings...

👉 I am interested in how to successfully use audit and VPD

👉 I want to talk about the performance issues with these technologies

Agenda

- 👉 The problem
- 👉 A structured solution to audit
- 👉 Reduce the audit problem
- 👉 A look at some simplistic performance testing
- 👉 VPD issues
- 👉 VPD simplistic performance testing
- 👉 Tuning the solution
- 👉 Conclusions

The perceived problem

- 👉 Why do people think Audit, FGA and VPD causes performance issues?
- 👉 Lack of experience of the technologies?
- 👉 Discussions on the net and newsgroups (There is little out there) – most negative, very few positive
- 👉 No real evidence as to performance problems
- 👉 The problem is “it depends”
- 👉 Most databases I have seen have the same audit issue
 - 👉 The customer is told or decides to enable audit
 - 👉 Little planning and design in advance
 - 👉 All application tables are audited
 - 👉 EFFECT = Data overload and performance problems
 - 👉 RESULT = Customer turns it all off!

Structured solutions for audit

 Design first

 Understand what is to be audited at a business level

 Audit data collection

 Reporting

 Alerting

 Management – archive / purge / future use

 Break the problem into smaller parts based on data that has to be collected

 Analyse for whom and when audit needs to be collected

 Analyse all data access to potential audit events – time / quantities

Reduce the audit problem (categorise)

 Based on requirements and design decide on solutions


 Limit the collection of audit data on a

 User basis

 Time basis

 Access basis

 Choose appropriate solutions based on these rules

 Use a risk-based approach to audit collection / process and use

 Does all audit collection cause a performance issue?

Does all audit kill performance

👉 First, let's talk about why we may want to audit

👉 Regulatory (SOX, PCI...)

👉 Financial reconciliation

👉 HMG requirements

👉 Internal requirements

👉 Some reasons to audit cannot be avoided – we must do them

👉 Some reasons to audit can be modified on a risk-based approach

👉 Some reasons to audit can be changed / re-specified - audit something else – up or down the business process

👉 Does it kill performance? – **it depends!**

Performance tuning

👉 There are two common elements to performance tuning

👉 Tune and make the existing code go faster

👉 Tune the algorithm first

👉 The second item is the interesting one for me

👉 The focus of this talk is not on lots of detailed performance tests but on tuning the design first to ensure


👉 The right audit tool / features are used

👉 The right audit events are defined and set


👉 The right audit data is collected

👉 Implementing other controls to reduce the need for auditing


Reduce the problem


 **E**mphasise: reduce the amount of audit that could have performance issues to the bare minimum or move the problem

 **B**ased on

 **O**nly audit what is needed at a business level based on risk

 **O**nly audit users / roles / time based

 **S**ome audit solutions can be used OLTP and turned off for batch processes or vice-versa

 **S**ome audit may be disabled during some time periods (this allows expensive audit for OLTP and none for batch)

Standard Audit solutions (horses for courses)

👉 Use the right solutions for the right purposes. Consider:

👉 Data to be captured

👉 Identification data – user ID, terminal, etc

👉 Before and after values

👉 Functionality of the solution

👉 Can it be based on user, role, columns, rows of data?

👉 Can it be disabled if necessary

👉 Writing to the file system for additional security / performance

Oracle standard audit solutions

👉 Standard database audit

👉 Triggers

👉 System triggers

👉 Fine Grained Audit

👉 Log files

👉 CDC / LogMiner / redo analysis

👉 Network Appliances – There are issues

👉 Custom solutions – Application based

👉 Who/when columns

👉 E-Business Suite, RLA, Sign-on, who/when

👉 More...

Oracle Standard Audit Solutions - comparison

	database	triggers	FGA	Row Level Audit	CDC	system triggers
Performance	8% - 200%	3% - 37%	3% - 300%	not tested	0%	not tested
before / after	No	Yes	Flashback / redo	Yes	Yes	N/A
Column level control	No	Yes	Yes	Yes	Yes	N/A
row level control	No	Yes	Yes	Edit	Possible	N/A
control by user	No	Yes	Yes	Yes	Possible	Yes
extend data capture	SQL Statement	Yes	Yes	Edit	Yes	Yes
read audit	Yes	No	Yes	No	No	N/A
write to file system	Yes	Yes	Yes	No	Yes	Yes

Some audit should be enabled anyway

👉 Whether there is a performance issue or not – I don't believe there is for some audit – some audit should always be enabled

👉 1g will default 24 database audit settings

👉 Logon audit – regulatory – memo22

👉 Unusual hours

👉 Shared connections

👉 Super user access

👉 Use of system privileges

👉 In the database

👉 In the application layer

Types of audit events

- 📌 **T**ypes of audit event need to be considered as these directly affect the solution chosen / the amount of data collected and when / the scope for tuning / creativity
- 📌 **C**onnection / privilege audit - use of privileges should not create performance issue as they shouldn't be used
- 📌 **S**ecurity configurations – again should not change
- 📌 **S**tatic and configuration data – U,I,D – OK, S - Issue
- 📌 **P**rocess and workflow audit - event auditing
- 📌 **B**usiness audit – the biggest potential for issues

Be Creative!

👉 **CDC** – no direct impact – hard to configure – extra hardware needed, bandwidth issues

👉 **Log miner** similar to CDC

👉 **Flashback** – useful but very time constrained

👉 **Network appliances** – there are issues with all

👉 **No direct performance issues**

👉 **Some do not support local connections**

👉 **Some store the data captured**

👉 **Even if capture local and remote, if packages are used then only the package call is seen by the appliance not the data access (if data is not returned to the client)**

Performance testing

- Example test for triggers

- Simple trigger

- Wanted to test

 - OLTP impact

 - Batch impact

 - Compared to no trigger

- Planning ahead to tuning

 - Use of When clause

 - Use of “OF” clause

 - Write to file system

Sample trigger – create a context

```
create or replace package audit_trig is
procedure set_context;
procedure set_off;
end;
/
```

```
create or replace package body audit_trig is
  lv_context constant varchar2(30):='PXF';
procedure set_context is
begin
  dbms_session.set_context(lv_context,'audit_trig','Y');
end;
procedure set_off is
begin
  dbms_session.set_context(lv_context,'audit_trig','N');
end;
end;
/
```

Sample Trigger – create the trigger

```
create or replace trigger pxf_t_u
after update on po.po_vendors
for each row
when
  (sys_context('PXF','audit_trig')='
  Y')
declare
  PRAGMA AUTONOMOUS_TRANSACTION;
begin
  insert into apps.po_shadow
  (
    date_time,
    userid,
    event_type,
    origin,
    vendor_name_old,
    vendor_name_new
  )
  values
  (
    sysdate,
    sys_context('USERENV',
                'CURRENT_USER'),
    'U',
    sys_context('USERENV',
                'IP_ADDRESS'),
    :old.vendor_name,
    :new.vendor_name);
  commit;
exception
  when others then
    null;
end;
/
```

Sample Trigger tests – no trigger enabled

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	5	0.00	0.00	0	0	0	0
Execute	6	0.03	0.03	0	0	1	3
Fetch	0	0.00	0.00	0	0	0	0
total	11	0.03	0.03	0	0	1	3

Misses in library cache during parse: 0

Elapsed times include waiting on following events:


Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	10	0.00	0.00
SQL*Net message from client	10	0.00	0.00
log file sync	1	0.05	0.05

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	8	0.00	0.00	0	0	0	0
Execute	1206	0.51	0.53	0	205	834	300
Fetch	1708	0.08	0.07	0	6132	0	1104
total	2922	0.59	0.61	0	6337	834	1404

Misses in library cache during parse: 0

Sample trigger – firing for OLTP

 37% impact

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	5	0.01	0.01	0	0	0	0
Execute	6	0.02	0.04	0	0	1	3
Fetch	0	0.00	0.00	0	0	0	0
total	11	0.03	0.05	0	0	1	3

Misses in library cache during parse: 1

Elapsed times include waiting on following events:

Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	10	0.00	0.00
SQL*Net message from client	10	0.00	0.00
log file sync	2	0.03	0.03

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	19	0.02	0.01	0	0	0	0
Execute	1322	0.71	0.69	0	205	1242	400
Fetch	1740	0.09	0.08	0	6212	0	1124
total	3081	0.82	0.80	0	6417	1242	1524

Misses in library cache during parse: 3

Misses in library cache during execute: 2

Sample trigger – conditionally disabled

 3% impact

```
*****
OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS
call      count      cpu      elapsed      disk      query      current      rows
-----
Parse      5         0.00      0.00         0          0          0          0
Execute    6         0.01      0.03         0          0          1          3
Fetch      0         0.00      0.00         0          0          0          0
-----
total     11         0.01      0.03         0          0          1          3
```

Misses in library cache during parse: 0

Elapsed times include waiting on following events:

Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	10	0.00	0.00
SQL*Net message from client	10	0.00	0.00
log file sync	1	0.04	0.04

```
OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS
call      count      cpu      elapsed      disk      query      current      rows
-----
Parse     11         0.01      0.00         0          0          0          0
Execute  1209       0.53      0.54         0         220        886        302
Fetch    1710       0.09      0.07         0        6135         0        1105
-----
total    2930       0.63      0.62         0        6355        886        1407
```

Misses in library cache during parse: 0

11g Improvements

- 👉 Ashwini Surpur announced at OOW that standard database audit is much faster
- 👉 He reported a 1 – 2% performance degradation for the TPCC benchmark with `audit_trail=DB` and some default audit settings
- 👉 Bryn Llewellyn announced that triggers are just faster in 11g. He quoted up to 25% faster for DML
- 👉 He also announced a new type of compound trigger. That is a before, after, row and statement all in one and it can retain PL/SQL global variable state between firings
- 👉 Oracle audit is on by default for 24 events

Fine Grained Audit

- 👉 Added to satisfy read audit requirements – SoX
- 👉 Likened to a select trigger
- 👉 Requires coding
- 👉 Hard to set up and get working – hard to debug
- 👉 DML support added in 10gR1
- 👉 Fires at the statement level
- 👉 Difficult to get before and after values
- 👉 Difficult to parse and use SQL statements and binds constructively
- 👉 Lots of bugs in early versions

FGA examples

```
begin
sys.dbms_fga.add_policy(
  object_schema => 'PO',
  object_name => 'PO_VENDORS',
  policy_name => 'PXF_TEST',
  audit_condition => ''Y' =
  sys_context('PXF','audit_trig')',
  audit_column => 'VENDOR_NAME',
  handler_schema => NULL,
  handler_module => NULL,
  enable => TRUE,
  statement_types => 'UPDATE',
  audit trail => dbms fga.db extended);
end;
/
```

Same context
as trigger was
used

FGA test – no policy

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	5	0.00	0.00	0	0	0	0
Execute	6	0.03	0.02	0	0	1	3
Fetch	0	0.00	0.00	0	0	0	0
total	11	0.03	0.03	0	0	1	3

Misses in library cache during parse: 0

Elapsed times include waiting on following events:

Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	10	0.00	0.00
SQL*Net message from client	10	0.00	0.00
log file sync	1	0.05	0.05

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	8	0.01	0.00	0	0	0	0
Execute	1206	0.55	0.55	0	205	834	300
Fetch	1708	0.07	0.08	0	6132	0	1104
total	2922	0.63	0.63	0	6337	834	1404

Misses in library cache during parse: 0

FGA test – creating audit - OLTP

 348% impact

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	5	0.04	0.03	0	0	0	0
Execute	6	1.20	2.42	2	6778	2489	3
Fetch	0	0.00	0.00	0	0	0	0
total	11	1.24	2.46	2	6778	2489	3

Misses in library cache during parse: 1

Elapsed times include waiting on following events:

Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	10	0.00	0.00
SQL*Net message from client	10	0.00	0.00
log file sync	1	0.04	0.04


OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	143	0.03	0.03	0	0	0	0
Execute	1349	1.64	2.98	2	7018	3030	411
Fetch	1965	0.09	0.12	2	6321	0	1337
total	3457	1.76	3.14	4	13339	3030	1748

Misses in library cache during parse: 5

Misses in library cache during execute: 4

FGA test - disabled

 13% impact

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	5	0.00	0.00	0	0	0	0
Execute	6	0.03	0.03	0	0	1	3
Fetch	0	0.00	0.00	0	0	0	0
total	11	0.03	0.03	0	0	1	3

Misses in library cache during parse: 0

Elapsed times include waiting on following events:

Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	10	0.00	0.00
SQL*Net message from client	10	0.00	0.00
log file sync	1	0.02	0.02

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	11	0.00	0.00	0	0	0	0
Execute	1209	0.60	0.60	0	220	897	302
Fetch	1710	0.05	0.08	0	6135	0	1105
total	2930	0.65	0.69	0	6355	897	1407

Misses in library cache during parse: 0

Virtual Private Database

- ✔ Supports static policies
- ✔ Don't access dual in policies
- ✔ Supports policy groups so can be effectively turned off
- ✔ Debugging is very hard
- ✔ Working out the predicate can be done - use
 - ✔ `$vpd_policy`
 - ✔ Set event 10730
 - ✔ Set event 10060

VPD performance testing – simple policy

```
create context vpd using scott.vpd_pkg;
begin
  dbms_ols.create_policy_group
  (object_schema=>'SCOTT',
   object_name => 'EMP',
   policy_group => 'VPD_OFF');
end;
/
begin
  dbms_ols.create_policy_group
  (object_schema => 'SCOTT',
   object_name => 'EMP',
   policy_group => 'VPD_ON');
  dbms_ols.add_grouped_policy
  (object_schema => 'SCOTT',
   object_name => 'EMP',
   policy_group => 'VPD_ON',
   policy_name => 'VPD_POL',
   function_schema => 'SCOTT',
   policy_function => 'VPD_PKG.GET_PREDICATE',
   update_check => FALSE,
   enable => TRUE,
   static_policy => TRUE);
end;
```

VPD code – Cont'd

```
begin
  dbms_ols.add_policy_context
  (object_schema => 'SCOTT',
  object_name => 'EMP',
  namespace => 'vpd',
  attribute => 'active_policy');
END;
/
create or replace package vpd_pkg is
procedure set_context;
procedure set_off;
  function get_predicate (object_schema in
  varchar2,object_name in varchar2) return
  varchar2;
end;
/
create or replace package body vpd_pkg is
  --
  lv_context constant varchar2(30):='vpd';
  --
  procedure set_context is
  begin
  dbms_session.set_context(lv_context,'act
  ive_policy','VPD_ON');
  end set_context;

  procedure set_off is
  begin
  dbms_session.set_context(lv_context,'a
  ctive_policy','VPD_OFF');
  end set_off;
  --
  function get_predicate (object_schema
  in varchar2,object_name in varchar2)
  return varchar2 is
  begin
  return (' deptno in (select
  deptno from dept where flag =
  'N')');
  end get_predicate;
End;
/
```

VPD – no policy

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	2	0.00	0.00	0	0	0	0
Execute	3	0.17	0.22	0	0	0	1
Fetch	0	0.00	0.00	0	0	0	0
total	5	0.17	0.22	0	0	0	1

Misses in library cache during parse: 0

Elapsed times include waiting on following events:


Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	4	0.00	0.00
SQL*Net message from client	4	0.00	0.00

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	1000	0.08	0.08	0	0	0	0
Fetch	1000	0.13	0.12	0	7000	0	14000
total	2001	0.21	0.20	0	7000	0	14000

Misses in library cache during parse: 0

VPD – protecting access

 100% impact

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	2	0.00	0.00	0	0	0	0
Execute	3	0.16	0.21	0	0	0	1
Fetch	0	0.00	0.00	0	0	0	0
total	5	0.16	0.21	0	0	0	1

Misses in library cache during parse: 0

Elapsed times include waiting on following events:


Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	4	0.00	0.00
SQL*Net message from client	4	0.00	0.00

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	1000	0.05	0.10	0	0	0	0
Fetch	1000	0.57	0.50	0	23000	0	12000
total	2001	0.62	0.60	0	23000	0	12000

Misses in library cache during parse: 0

VPD – enabled but not firing

 No impact

```
*****
OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS
call      count      cpu      elapsed      disk      query      current      rows
-----
Parse      2      0.00      0.00      0      0      0      0
Execute    3      0.16      0.23      0      0      0      1
Fetch      0      0.00      0.00      0      0      0      0
-----
total      5      0.16      0.23      0      0      0      1
```

Misses in library cache during parse: 0

Elapsed times include waiting on following events:

Event waited on	Times Waited	Max. Wait	Total Waited
SQL*Net message to client	4	0.00	0.00
SQL*Net message from client	4	0.00	0.00

```
OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS
call      count      cpu      elapsed      disk      query      current      rows
-----
Parse      1      0.00      0.00      0      0      0      0
Execute   1000      0.06      0.10      0      0      0      0
Fetch     1000      0.12      0.11      0      7000      0     14000
-----
total     2001      0.18      0.21      0      7000      0     14000
```

Misses in library cache during parse: 1

VPD compared with FGA

- 👉 FGA does not include as much functionality as VPD
- 👉 FGA cannot disable the core write to FGA_LOG\$
- 👉 VPD has no core write to the file system or database
- 👉 Would be useful to have FGA and VPD in one policy – can do this for audit
- 👉 VPD could be used for audit
- 👉 FGA doesn't support policy groups or policy contexts
- 👉 VPD seems to be faster

Tuning the solutions

- 👉 Write to the file system instead of the database
- 👉 Simplify any audit code
- 👉 Ensure audit fires only for users / columns / rows necessary
- 👉 Use static data
- 👉 Limit database access in policy functions
- 👉 Simplify the predicate and avoid excessively changing the optimizer path

Audit information

Books:

 Knox - Effective Oracle Databases 10g Security by Design (Osborne Oracle Press) – ISBN - 0072231300

 Arup Nanda - Oracle PL/SQL for DBAs – ISBN 0596005873

 Therioult, Henney - Oracle Security – ISBN 1565924509

Papers

 An Introduction to Simple Oracle Auditing - <http://www.securityfocus.com/infocus/1689>

 Oracle Auditing - <http://www.oracle-base.com/articles/8i/Auditing.php>

Test and test again

👉 Testing is very subjective to the

👉 Application structure – including the SQL used

👉 Application use, OLTP, DSS, Batch based...

👉 Physical and logical architecture

👉 Hardware specifications

👉 OLTP access can often be faster with VPD in place due to often reduced result sets

👉 Relying on other peoples studies is pointless

👉 Design and scope first

👉 Test on a real system

👉 Test on real data and quantities

Conclusions

- 👉 Performance impact depends on the design
- 👉 Design to capture the audit that is necessary, design for speed where it counts
- 👉 Be creative
- 👉 Testing is very subjective
- 👉 The tests should be for batch users and OLTP users and any other types of users
- 👉 Aim to reduce the problem
- 👉 If there is a performance issue then its better to have it on a smaller number of tables
- 👉 Don't abandon audit because of perceived issues, TEST

Questions and Answers

👉 Any Questions, please ask

👉 Later?

👉 Contact me via email peter.finnigan@siemens.com

👉 Or via my website <http://www.petefinnigan.com>



www.siemens.co.uk/insight

+44 (0)1932 241000

Insight Consulting

Siemens Enterprise Communications Limited

**Security, Compliance, Continuity
and Identity Management**

SIEMENS



Choose with confidence
use with ease



INVESTOR IN PEOPLE



013