PeteFinnigan.com Limited
Oracle Security

Building Practical Audit Trails

Design easy to implement and use audit trails

Legal Notice

Oracle Database Security Presentation

Published by
PeteFinnigan.com Limited
9 Beech Grove
Acomb
York
England, YO26 5LD

Pete Finnigan – Who Am I?

- Oracle Security specialist and researcher
- CEO and founder of PeteFinnigan.com Limited in February 2003
- Writer of the longest running Oracle security blog
- Author of the Oracle Security step-by-step guide and more recently "Oracle Expert Practices"
- Member of the OakTable
- Speaker at various conferences
  - UKOUG, PSOUG, BlackHat, more..
- Published many times, see
  - www.petefinnigan.com for links
- Influenced industry standards
  - And governments

Agenda

- History of this talk
- Design Must Come first
- What is practical?
- Background / Possible Solutions
- Solutions and Examples

History; 12c

- I wrote this presentation back in 2012 and presented it just once at a SIG
- I subsequently designed policy based auditing and VPS and a simple firewall
- This then became the basis of a one day class on the same subject
- 12c came along and unified audit was part of 12c. I was not in the beta
- The material here is relevant for 12c either using standard audit or unified audit
- The same nuances apply whether core audit or unified
- I am going to focus at a high level but also use core audit and not 12c Unified audit

Design Comes First – No Matter the solution

- Before we get started implementing
- Design must be the first step
- The solution implements the design
- Therefore until you know the design you cannot specify the right solution – right?
- The solution could be “free” or “commercial” solution or even a combination of both
- So often people buy products and implement out of the box with no internal requirements

“What do I want to know?”

- Start with “what do I want to know?”
  - Risk based
  - Based on regulations
  - Based on business needs
- Then design technical solutions to implement these requirements
- Include
  - Sizing, performance, tech data collect, transmit/ transfer if relevant, raw trail storage, reports, alerts, management, issues management, more...

So What Is Practical In This Context?

- Quick to implement
- Cheap in software license / time
- Easy to maintain and extend
- Easy to configure in the first place
- Provides value vs risk
- On-going support
- Easy customisation

Fundamentals

- Do we satisfy auditors or compromise?
  - Risk vs cost (implement and TCO/ROI)
- Keep the raw trail or the reports?
- Trail to be kept off the server or local?
- Size of the storage required?
- Performance (depends on actions captured and design decisions)
- Re-Active “vs” Pro-Active audit

The Case For Free Core Solutions

- The free core solutions are worth using
- Simple to design
  - We may have to be creative
- DBA can implement and maintain
- Should be easy to extend
- If database used (aud$) reports are simple
- Can use database to manage

Possibilities to Audit an Oracle Database?

- Options – at a very high level
  - DAM – Database Activity Monitoring
  - IDS – Intrusion Detection System
  - IPS – Intrusion Prevention System
  - VPS – Virtual Patching Systems
  - Audit Log Monitoring
  - Centralised Audit – Audit Vault
  - Native database audit / OS audit features
- Solutions can be
  - Network / host
  - Hardware / software
  - Both / All

Address The Elephant In The Room

- It is: **Performance of database audit**
- Storage to a lesser extent; we need to consider
  - Do we store results or alerts/reports – regulations will impact this choice
- Design with performance in mind (often we can catch an issue (attack) without creating an extensive trail)
- Capture non-standard access – no performance issue
- Because of perceived performance problems sites do nothing in the core; yet use DML triggers in applications with horrendous performance results

Not scientific; high level; not every bolt and nut
Yes / No maybe good/Bad!

Options Compared – Commercial / Free

Option	Commercial	Free (Database)
0-Day Detection	Yes (open to debate)	No
Detailed Reports	Yes	No
Virtual Patch	Yes (most vendors)	No
Out of the box protection	Yes	No – well sort of
Fast deployment	No / Yes (depends)	Yes
Alerts and Escalation	Yes	No
Storage	Yes / No (depends)	Yes
Management of rules/reports/storage	Yes	No
Management of issues located	Yes	No
Dashboard – Plasma screen!!!	Yes	No
Real time monitoring	Yes	Yes
Security of Audit	Yes	Maybe
Support	Yes	No
License cost	Yes	No
Extra Hardware / Software	Yes	Maybe

Mostly free audit can do the job but we need to design and build the infrastructure. Perceived risk because it is internal

Options Compared – Commercial / Free

Option	Commercial	Free (Database)
0-Day Detection	Yes (open to debate)	No **(We can implement)**
Detailed Reports	Yes	No **(We can write them)**
Virtual Patch	Yes (most vendors)	No **(We can create simple)**
Out of the box protection	Yes	No – well sort of
Fast deployment	No / Yes (depends)	Yes
Alerts and Escalation	Yes	No **(We can write and create)**
Storage	Yes / No (depends)	Yes
Management of rules/reports/storage	Yes	No **(Harder to make simple)**
Management of issues located	Yes	No **(harder again but can do)**
Dashboard – Plasma screen!!!	Yes	No **(can create simple with SQL)**
Real time monitoring	Yes	Yes
Security of Audit	Yes	Maybe **(not by default)**
Support	Yes	No
License cost	Yes	No
Extra Hardware / Software	Yes	Maybe

Local / Remote Comparison

Option / Issue	Commercial	Free
Trail storage	Remote	Local or Remote
Rule storage	Remote (but can be part local)	Local
Collection of raw date	Remote (can be local)	Local
Process / analysis	Remote	None
Alerts / issues	Remote	None

Comparison Conclusions

- Commercial gives some value “**Out of the box**”
- Commercial gives management, alerts, rules “**Out of the box**”
- Free core database options have no extra license cost; commercial can be expensive
  - Both would have ongoing costs to configure and maintain
- There are areas where commercial solutions have issues; network sniffing can miss something
- Both need to be customised – so does core solutions
- The same design process should be followed for any solution – “commercial or free”

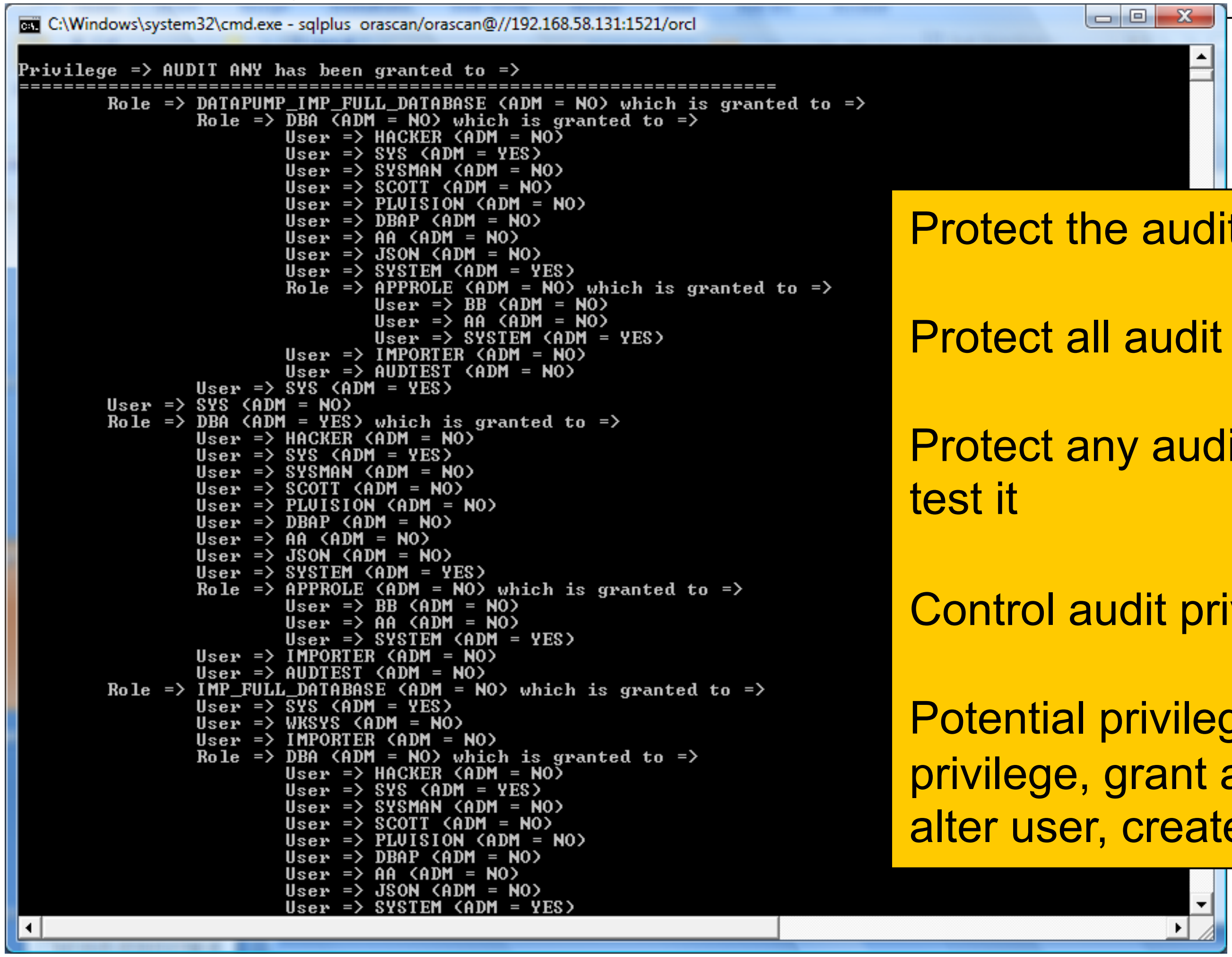
Solution Requirements (assuming Core Database)

- Privileged user (SYSDBA/SYSOPER/SYSASM) access audit
- Startup/shutdown - Mandatory audit
- System activity - Logs – listener, alert,...
- Capture privilege use (DDL)
- Third party access and actions – proxy audit
- Data access / change
- Audit security controls
- Audit the audit itself
- Static data and configuration data
- Connections by all users
- Attacks (CPU's, 0-days...)
- **All of these should not cause performance or functional issues**

Mechanical Process To Walk Through

- Take “What do I want to know” (Rules for set up)
- Design and deploy settings needed (+ code)
- Filter results / store / protect audit data and audit settings
- Produce reports – general reports
- Send alerts – These are specific “I **really** want to know”
- Manage raw data
- Manage extracted summary data
- Understand risk in solution choices – local vs remote

Security of Audit



C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/orcl

```
Privilege => AUDIT ANY has been granted to =>
====================================================================
        Role => DATAPUMP_IMP_FULL_DATABASE (ADM = NO) which is granted to =>
                Role => DBA (ADM = NO) which is granted to =>
                        User => HACKER (ADM = NO)
                        User => SYS (ADM = YES)
                        User => SYSMAN (ADM = NO)
                        User => SCOTT (ADM = NO)
                        User => PLVISION (ADM = NO)
                        User => DBAP (ADM = NO)
                        User => AA (ADM = NO)
                        User => JSON (ADM = NO)
                        User => SYSTEM (ADM = YES)
                        Role => APPROLE (ADM = NO) which is granted to =>
                                User => BB (ADM = NO)
                                User => AA (ADM = NO)
                                User => SYSTEM (ADM = YES)
                        User => IMPORTER (ADM = NO)
                        User => AUDTEST (ADM = NO)
                User => SYS (ADM = YES)
        User => SYS (ADM = NO)
        Role => DBA (ADM = YES) which is granted to =>
                User => HACKER (ADM = NO)
                User => SYS (ADM = YES)
                User => SYSMAN (ADM = NO)
                User => SCOTT (ADM = NO)
                User => PLVISION (ADM = NO)
                User => DBAP (ADM = NO)
                User => AA (ADM = NO)
                User => JSON (ADM = NO)
                User => SYSTEM (ADM = YES)
                Role => APPROLE (ADM = NO) which is granted to =>
                        User => BB (ADM = NO)
                        User => AA (ADM = NO)
                        User => SYSTEM (ADM = YES)
                User => IMPORTER (ADM = NO)
                User => AUDTEST (ADM = NO)
        Role => IMP_FULL_DATABASE (ADM = NO) which is granted to =>
                User => SYS (ADM = YES)
                User => WKSYS (ADM = NO)
                User => IMPORTER (ADM = NO)
                Role => DBA (ADM = NO) which is granted to =>
                        User => HACKER (ADM = NO)
                        User => SYS (ADM = YES)
                        User => SYSMAN (ADM = NO)
                        User => SCOTT (ADM = NO)
                        User => PLVISION (ADM = NO)
                        User => DBAP (ADM = NO)
                        User => AA (ADM = NO)
                        User => JSON (ADM = NO)
                        User => SYSTEM (ADM = YES)
```

Protect the audit trail with audit - next

Protect all audit trails – limit access

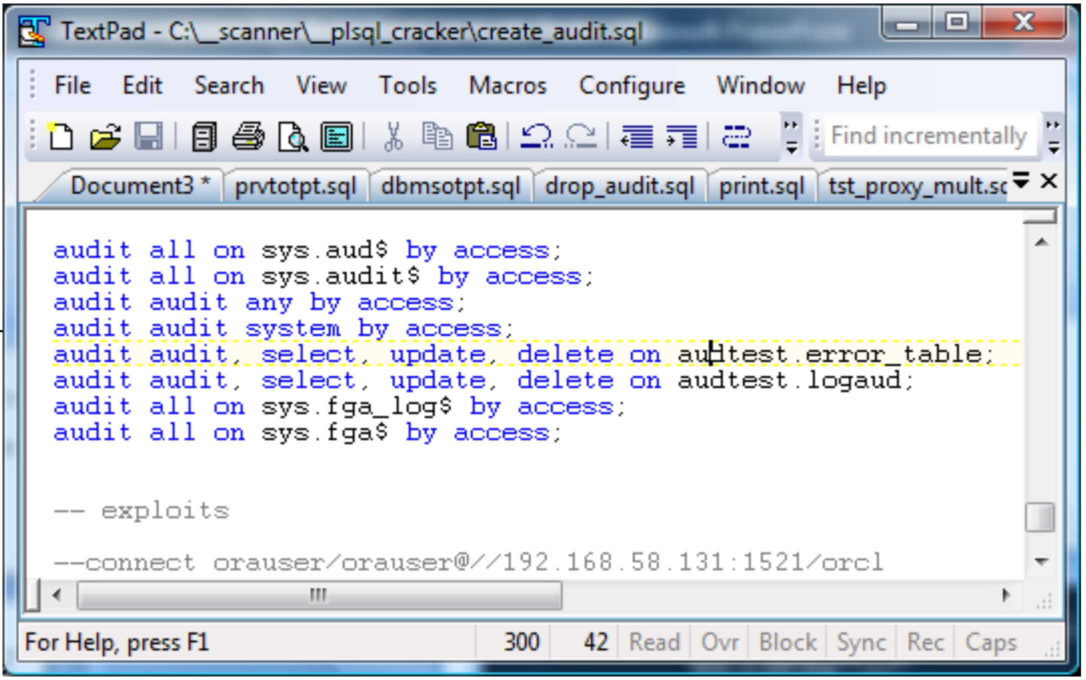
Protect any audit functionality used – test it

Control audit privileges granted – test

Potential privileges – grant any privilege, grant any object privilege, alter user, create any procedure…

PeteFinnigan.com Limited
Oracle Security

Audit of Audit



TextPad - C:_scanner_plsql_cracker\create_audit.sql
File Edit Search View Tools Macros Configure Window Help
Find incrementally
Document3 * prvtotpt.sql dbmsotpt.sql drop_audit.sql print.sql tst_proxy_mult.sc
For Help, press F1 300 42 Read Ovr Block Sync Rec Caps

```
audit all on sys.aud$ by access;
audit all on sys.audit$ by access;
audit audit any by access;
audit audit system by access;
audit audit, select, update, delete on audtest.error_table;
audit audit, select, update, delete on audtest.logaud;
audit all on sys.fga_log$ by access;
audit all on sys.fga$ by access;


-- exploits

--connect orauser/orauser@//192.168.58.131:1521/orcl
```

- Audit audit trail
- Audit FGA
- Audit triggers used
- Audit the audit system privileges
- Audit alter session ,system, database
- Audit shadow tables used – if any
- Identify all storage used, controls, PL/SQL functions and audit them as well

Audit of Security Controls

- Use audit facilities to audit security controls implemented in the database
- Identify all security controls that are enabled
  - VPD, system privileges, roles, secure application roles, parameters, schemas
  - Data level controls and grants
- Enable audit controls on the security controls identified and privileges used
- The solutions can be varied but layered i.e.
  - E.g. : Parameter change: Audit ALTER SESSION, ALTER SYSTEM and use a system startup/shutdown trigger

Customising Identity Example

```
SQL> select client_identifier from v$session
  2  where sid=(select distinct sid from v$mystat);
CLIENT_IDENTIFIER
----------------------------------------------------------


SQL> exec dbms_session.set_identifier('Hack the planet!');
PL/SQL procedure successfully completed.
SQL> select client_identifier from v$session
  2  where sid=(select distinct sid from v$mystat);
CLIENT_IDENTIFIER
----------------------------------------------------------
Hack the planet!
SQL> alter user orascan identified by orascan;
User altered.
SQL> select client_id from dba_audit_trail
  2  where sessionid=1854567;
CLIENT_ID
--------------------------------------------------
Hack the planet!
```

Each user must have an identity that can be relied upon

Must be set on logon (before logon ideally)

Blank identity is suspicious

Needs to be set to a trusted non-guessable value by the client (application not your customer)

The value shown here is the only modifiable field that permeates to core audit

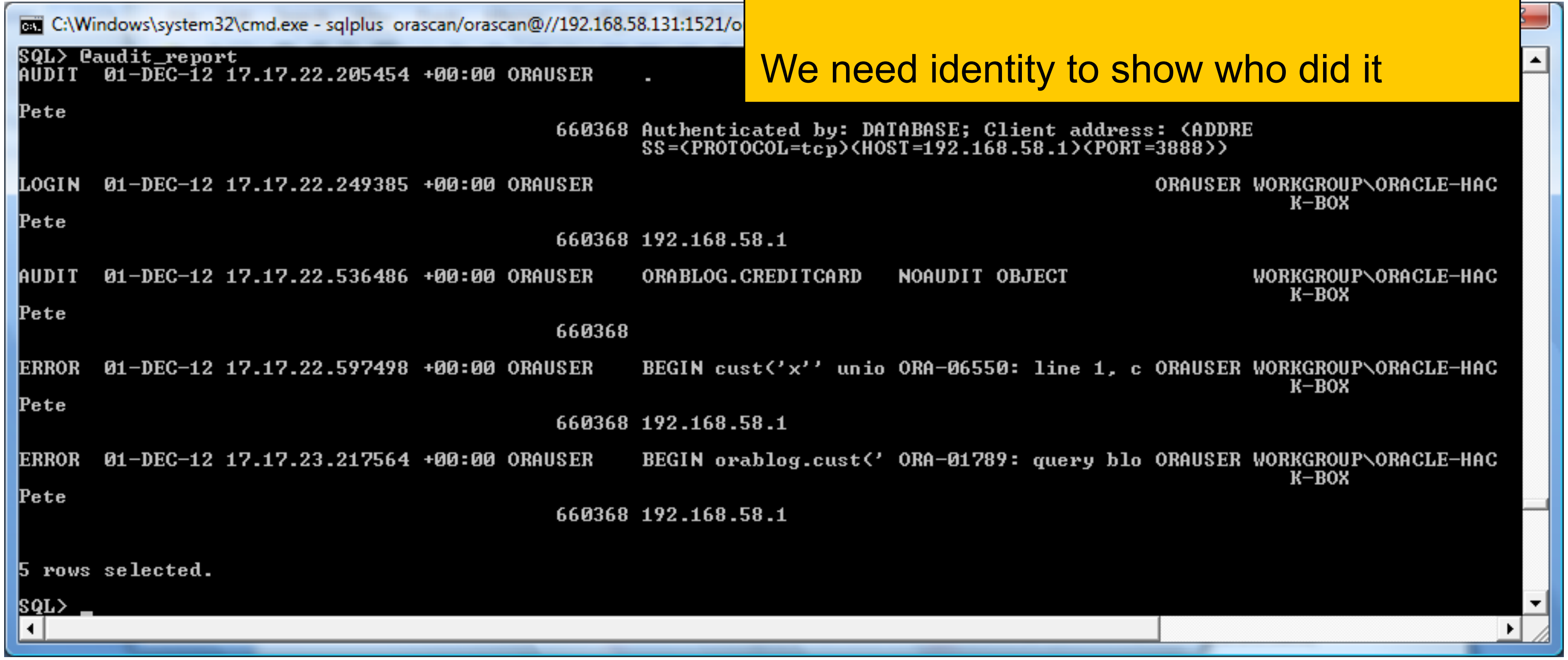
PeteFinnigan.com Limited
Oracle Security

Correlation

We can see correlation between core audit, logon audit, error audit, system triggers, custom code…

We could extend to others such as FGA, logs, more

We need identity to show who did it



C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/o

```
SQL> @audit_report
AUDIT   01-DEC-12 17.17.22.205454 +00:00 ORAUSER    .

Pete
                                                   660368 Authenticated by: DATABASE; Client address: (ADDRE
                                                          SS=(PROTOCOL=tcp)(HOST=192.168.58.1)(PORT=3888))

LOGIN   01-DEC-12 17.17.22.249385 +00:00 ORAUSER                                              ORAUSER WORKGROUP\ORACLE-HAC
                                                                                                          K-BOX
Pete
                                                   660368 192.168.58.1

AUDIT   01-DEC-12 17.17.22.536486 +00:00 ORAUSER    ORABLOG.CREDITCARD   NOAUDIT OBJECT               WORKGROUP\ORACLE-HAC
                                                                                                          K-BOX
Pete
                                                   660368

ERROR   01-DEC-12 17.17.22.597498 +00:00 ORAUSER    BEGIN cust('x'' unio ORA-06550: line 1, c ORAUSER WORKGROUP\ORACLE-HAC
                                                                                                          K-BOX
Pete
                                                   660368 192.168.58.1

ERROR   01-DEC-12 17.17.23.217564 +00:00 ORAUSER    BEGIN orablog.cust(' ORA-01789: query blo ORAUSER WORKGROUP\ORACLE-HAC
                                                                                                          K-BOX
Pete
                                                   660368 192.168.58.1


5 rows selected.

SQL> _
```

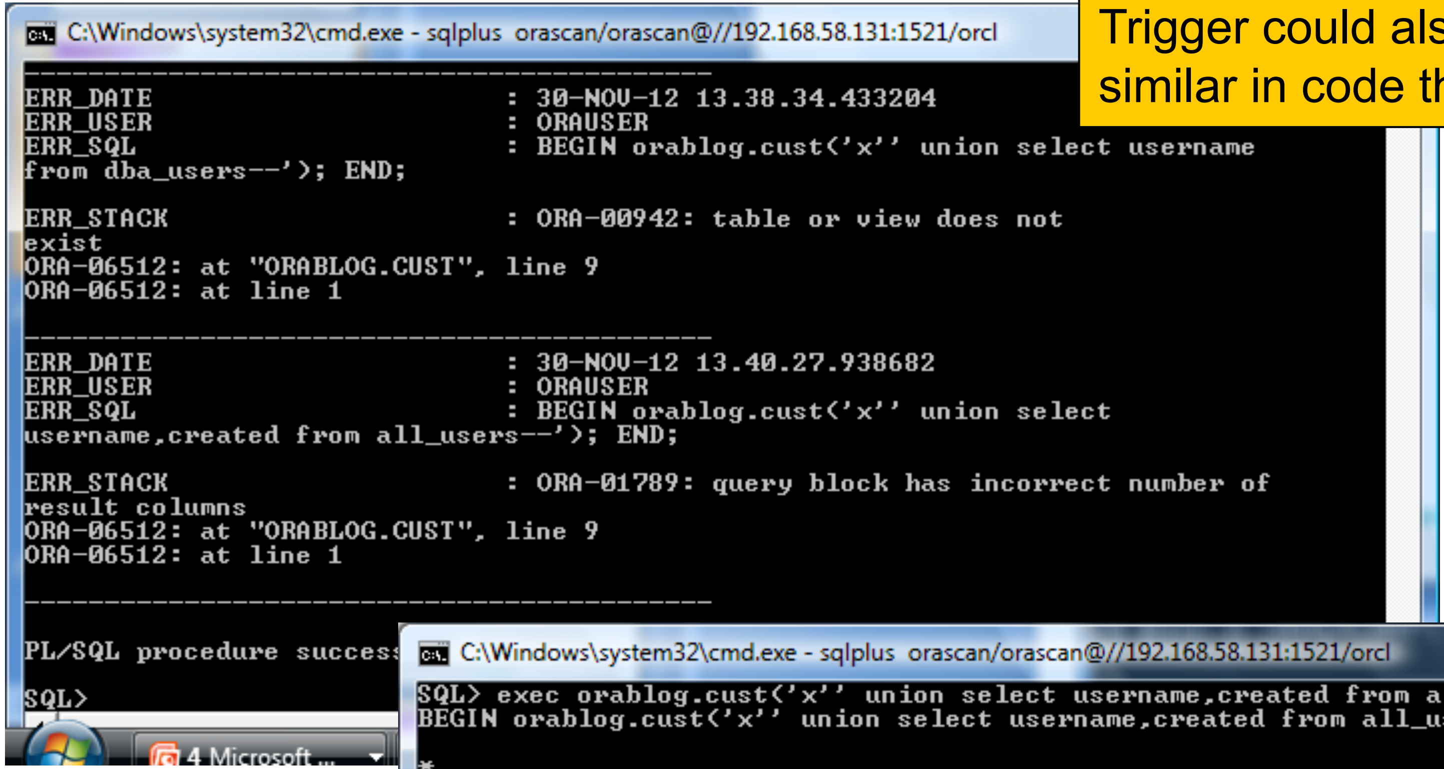
PeteFinnigan.com Limited
Oracle Security

SQL Injection and 0-Days - 1

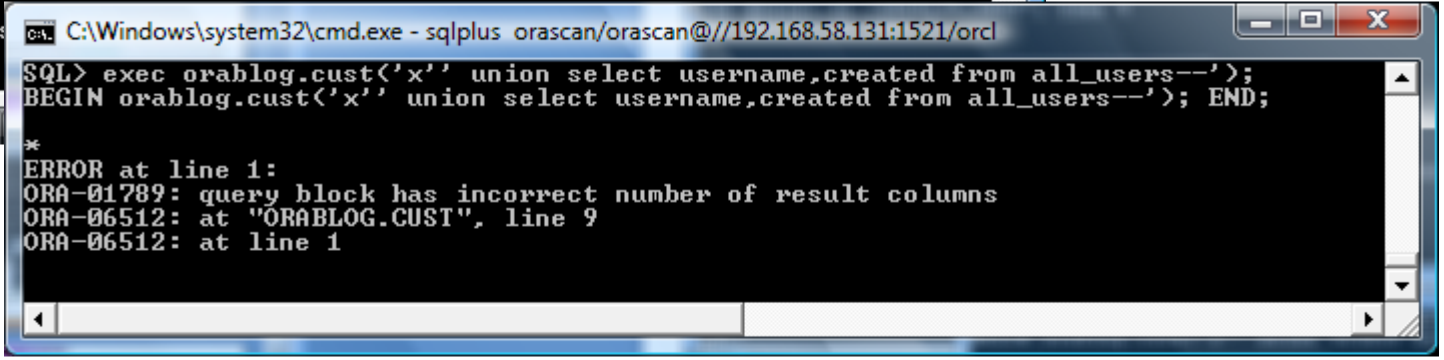
Driven by simple DDL system error trigger

Audit table needs to be protected

Trigger could also detect '--' or similar in code that is caught

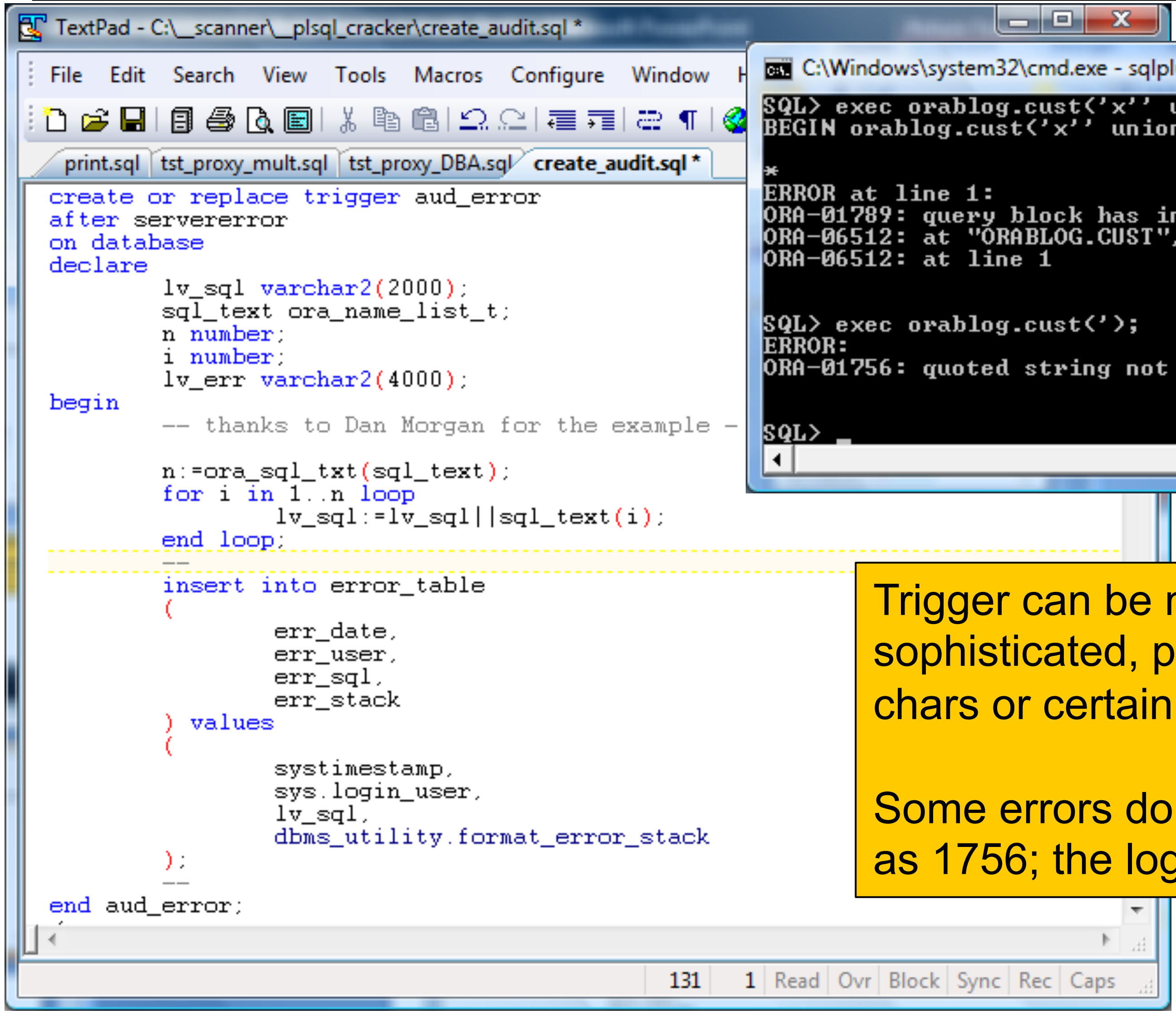

C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/orcl

```
-------------------------------------------
ERR_DATE                      : 30-NOV-12 13.38.34.433204
ERR_USER                      : ORAUSER
ERR_SQL                       : BEGIN orablog.cust('x'' union select username
from dba_users--'); END;

ERR_STACK                     : ORA-00942: table or view does not
exist
ORA-06512: at "ORABLOG.CUST", line 9
ORA-06512: at line 1

-------------------------------------------
ERR_DATE                      : 30-NOV-12 13.40.27.938682
ERR_USER                      : ORAUSER
ERR_SQL                       : BEGIN orablog.cust('x'' union select
username,created from all_users--'); END;

ERR_STACK                     : ORA-01789: query block has incorrect number of
result columns
ORA-06512: at "ORABLOG.CUST", line 9
ORA-06512: at line 1

-------------------------------------------

PL/SQL procedure success

SQL>
```

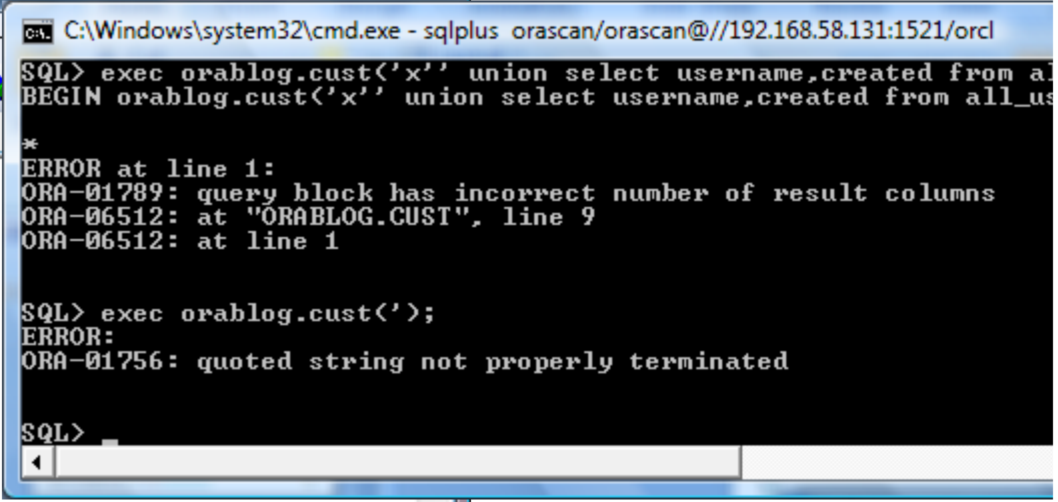

C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/orcl

```
SQL> exec orablog.cust('x'' union select username,created from all_users--');
BEGIN orablog.cust('x'' union select username,created from all_users--'); END;

*
ERROR at line 1:
ORA-01789: query block has incorrect number of result columns
ORA-06512: at "ORABLOG.CUST", line 9
ORA-06512: at line 1
```

SQL Injection and 0-Days - 2


TextPad - C:_scanner_plsql_cracker\create_audit.sql *
File Edit Search View Tools Macros Configure Window
print.sql tst_proxy_mult.sql tst_proxy_DBA.sql create_audit.sql *
131 1 Read Ovr Block Sync Rec Caps

```
create or replace trigger aud_error
after servererror
on database
declare
        lv_sql varchar2(2000);
        sql_text ora_name_list_t;
        n number;
        i number;
        lv_err varchar2(4000);
begin
        -- thanks to Dan Morgan for the example -

        n:=ora_sql_txt(sql_text);
        for i in 1..n loop
                lv_sql:=lv_sql||sql_text(i);
        end loop;
        --
        insert into error_table
        (
                err_date,
                err_user,
                err_sql,
                err_stack
        ) values
        (
                systimestamp,
                sys.login_user,
                lv_sql,
                dbms_utility.format_error_stack
        );
        --
end aud_error;
/
```


C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/orcl

```
SQL> exec orablog.cust('x'' union select username,created from al
BEGIN orablog.cust('x'' union select username,created from all_us

*
ERROR at line 1:
ORA-01789: query block has incorrect number of result columns
ORA-06512: at "ORABLOG.CUST", line 9
ORA-06512: at line 1


SQL> exec orablog.cust(');
ERROR:
ORA-01756: quoted string not properly terminated


SQL> _
```

Trigger can be much more sophisticated, parse out '--' or special chars or certain views or...

Some errors do not fire the trigger such as 1756; the logging is different

Virtual Patching

- Detects an attack to an un-patched system and blocks the attack
- An error trigger can be extended

```
If(ora_server_error(i)='1789') then
  log the event
  kill the session
```

- Detect use of vulnerable package – cpu fixed, dangerous packages (File, OS, Network...)

```
Audit all on utl_file
Audit all on xdb.xdb_pitrig_pkg
```

- Or create dummy package to allow actions to be blocked

Virtual Patch Dummy

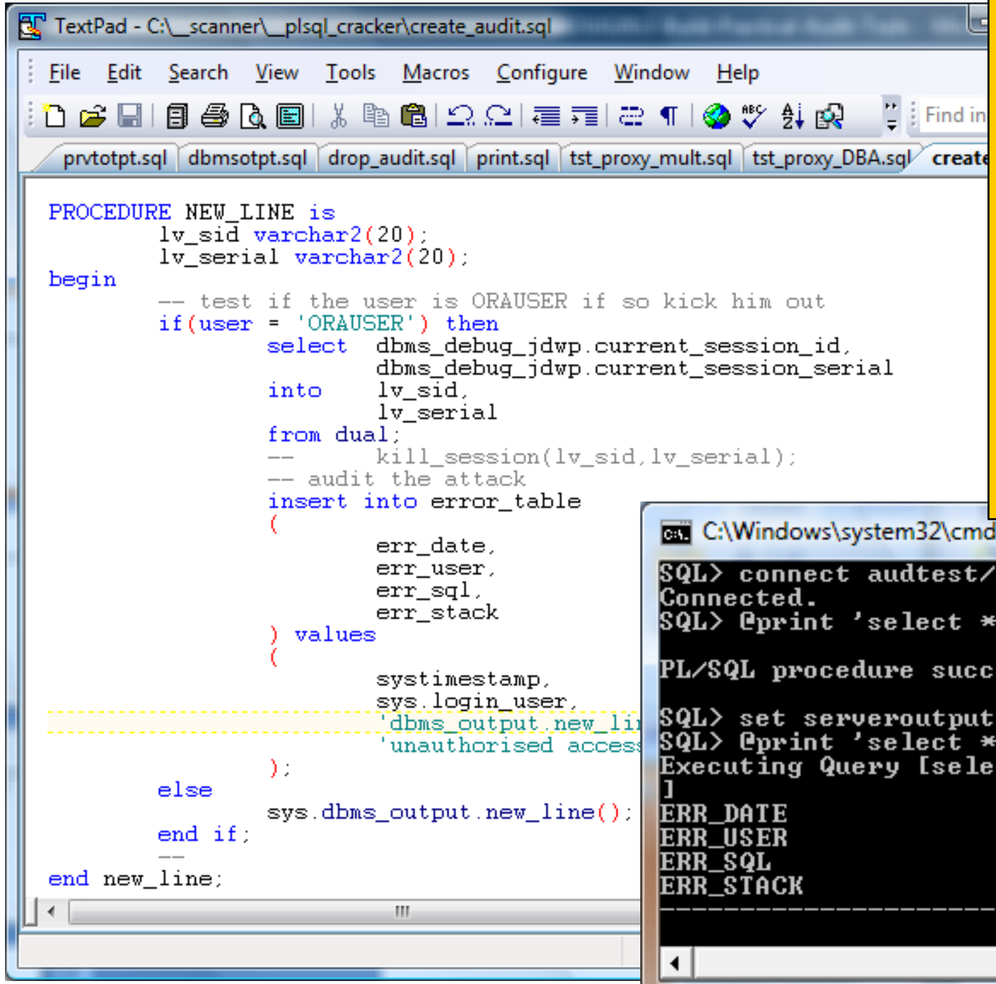
Create a copy of a vulnerable package from a CPU or other attack.

Call through to the real package

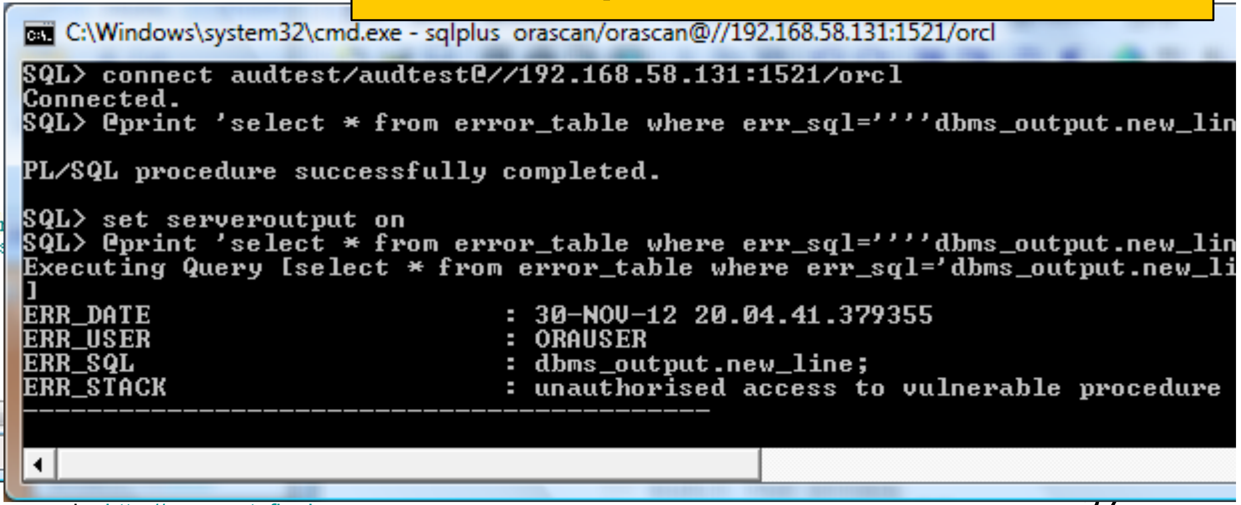
Implement context based test for attacks and log or kill

Kill is not simple as cannot kill current session but possible via job or daemon. Can also take other actions, lock, rollback...

Murder is possible, suicide is not

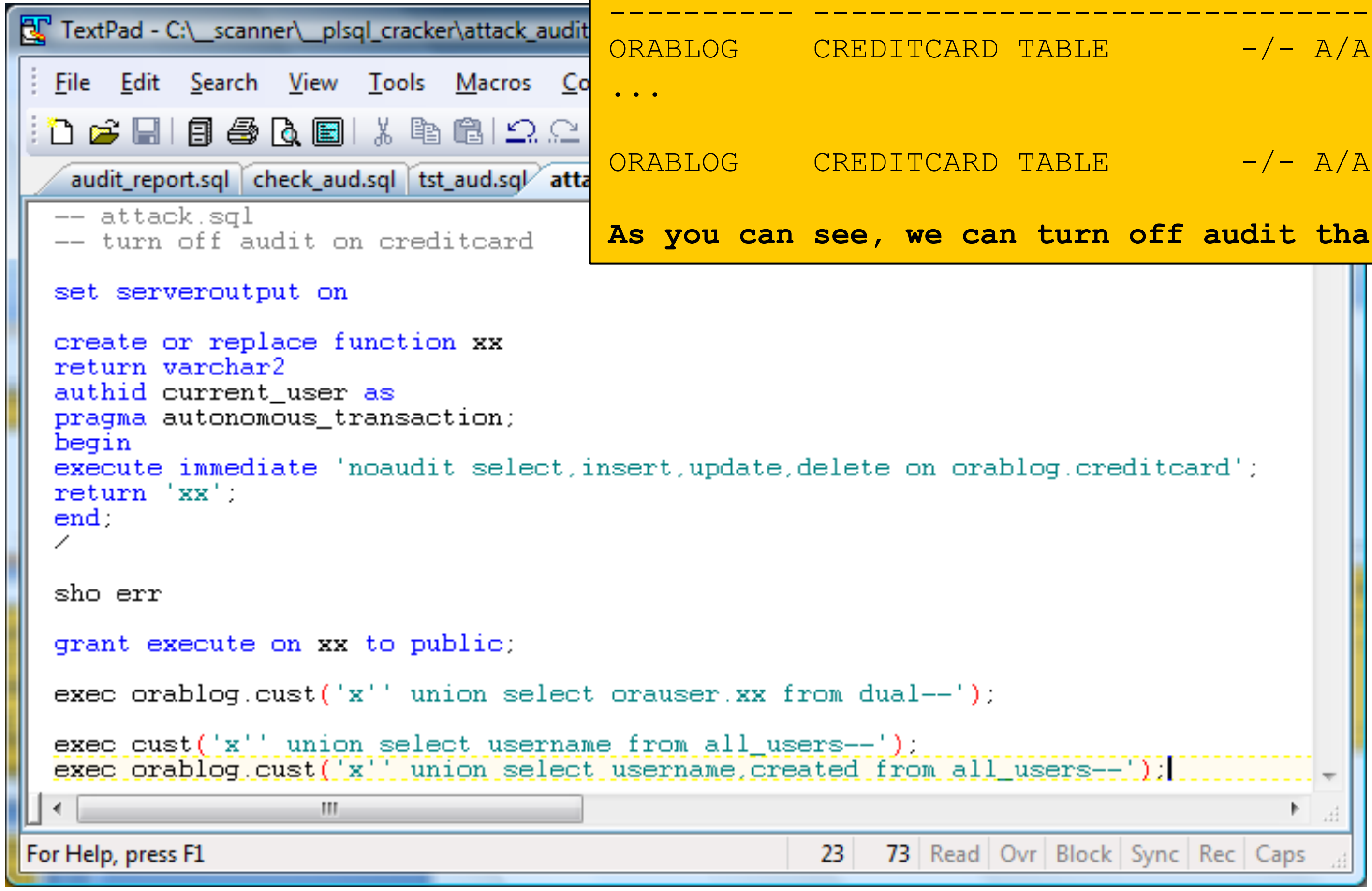

TextPad - C:_scanner_plsql_cracker\create_audit.sql
File Edit Search View Tools Macros Configure Window Help
prvtotpt.sql dbmsotpt.sql drop_audit.sql print.sql tst_proxy_mult.sql tst_proxy_DBA.sql create

```
PROCEDURE NEW_LINE is
        lv_sid varchar2(20);
        lv_serial varchar2(20);
begin
        -- test if the user is ORAUSER if so kick him out
        if(user = 'ORAUSER') then
                select  dbms_debug_jdwp.current_session_id,
                        dbms_debug_jdwp.current_session_serial
                into    lv_sid,
                        lv_serial
                from dual;
                --      kill_session(lv_sid,lv_serial);
                -- audit the attack
                insert into error_table
                (
                        err_date,
                        err_user,
                        err_sql,
                        err_stack
                ) values
                (
                        systimestamp,
                        sys.login_user,
                        'dbms_output.new_li
                        'unauthorised access
                );
        else
                sys.dbms_output.new_line();
        end if;
        --
end new_line;
```


C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/orcl

```
SQL> connect audtest/audtest@//192.168.58.131:1521/orcl
Connected.
SQL> @print 'select * from error_table where err_sql=''''dbms_output.new_lin

PL/SQL procedure successfully completed.

SQL> set serveroutput on
SQL> @print 'select * from error_table where err_sql=''''dbms_output.new_lin
Executing Query [select * from error_table where err_sql='dbms_output.new_li
]
ERR_DATE                      : 30-NOV-12 20.04.41.379355
ERR_USER                      : ORAUSER
ERR_SQL                       : dbms_output.new_line;
ERR_STACK                     : unauthorised access to vulnerable procedure
-------------------------------------------
```

Attack Audit Trails

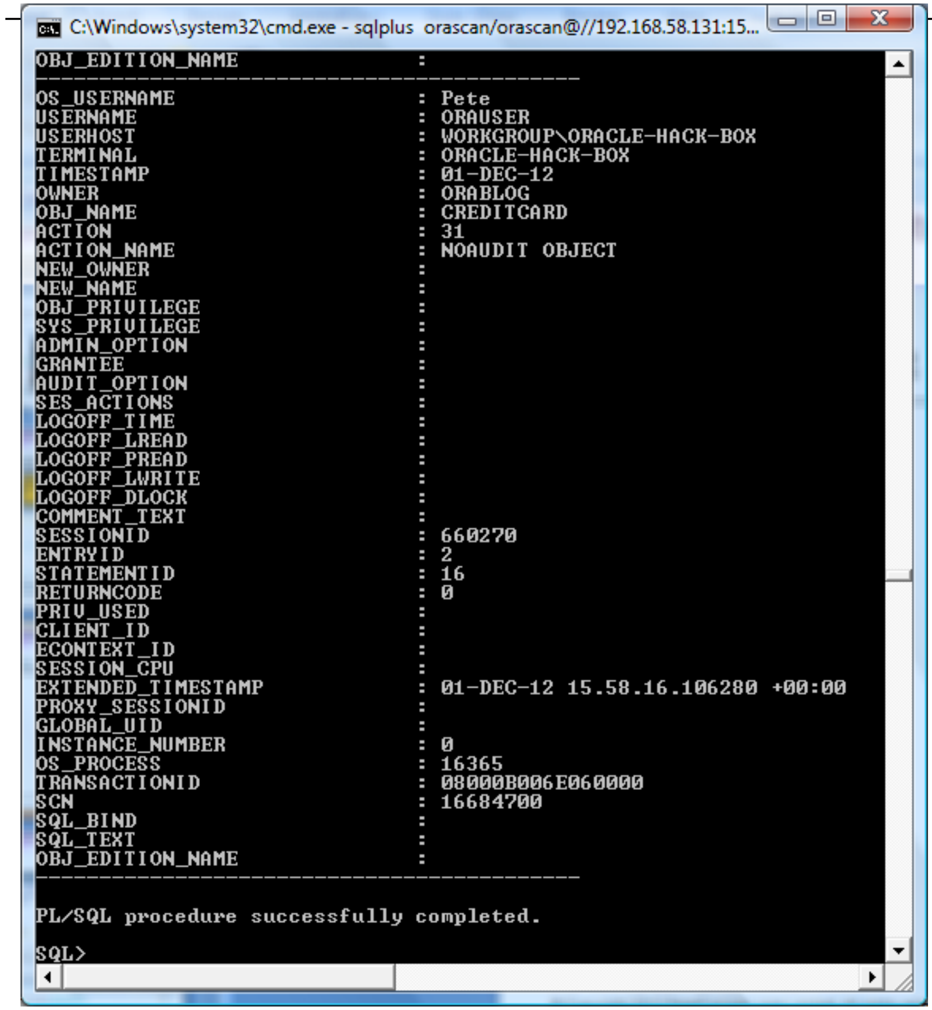

TextPad - C:_scanner_plsql_cracker\attack_audit
File Edit Search View Tools Macros Co
audit_report.sql check_aud.sql tst_aud.sql atta
For Help, press F1 23 73 Read Ovr Block Sync Rec Caps

```
-- attack.sql
-- turn off audit on creditcard

set serveroutput on

create or replace function xx
return varchar2
authid current_user as
pragma autonomous_transaction;
begin
execute immediate 'noaudit select,insert,update,delete on orablog.creditcard';
return 'xx';
end;
/

sho err

grant execute on xx to public;

exec orablog.cust('x'' union select orauser.xx from dual--');

exec cust('x'' union select username from all_users--');
exec orablog.cust('x'' union select username,created from all_users--');
```

```
OWNER      OBJECT     OBJECT_TYPE ALT AUD COM DEL GRA IND INS
---------- ---------------------------------- --- --- --- ---
ORABLOG    CREDITCARD TABLE       -/- A/A -/- A/A -/- -/- A/A
...

ORABLOG    CREDITCARD TABLE       -/- A/A -/- -/- -/- -/- -/- -/
```

As you can see, we can turn off audit that is enabled

Attack on Audit 2



C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:15...

```
OBJ_EDITION_NAME              :
------------------------------------------------
OS_USERNAME                   : Pete
USERNAME                      : ORAUSER
USERHOST                      : WORKGROUP\ORACLE-HACK-BOX
TERMINAL                      : ORACLE-HACK-BOX
TIMESTAMP                     : 01-DEC-12
OWNER                         : ORABLOG
OBJ_NAME                      : CREDITCARD
ACTION                        : 31
ACTION_NAME                   : NOAUDIT OBJECT
NEW_OWNER                     :
NEW_NAME                      :
OBJ_PRIVILEGE                 :
SYS_PRIVILEGE                 :
ADMIN_OPTION                  :
GRANTEE                       :
AUDIT_OPTION                  :
SES_ACTIONS                   :
LOGOFF_TIME                   :
LOGOFF_LREAD                  :
LOGOFF_PREAD                  :
LOGOFF_LWRITE                 :
LOGOFF_DLOCK                  :
COMMENT_TEXT                  :
SESSIONID                     : 660270
ENTRYID                       : 2
STATEMENTID                   : 16
RETURNCODE                    : 0
PRIV_USED                     :
CLIENT_ID                     :
ECONTEXT_ID                   :
SESSION_CPU                   :
EXTENDED_TIMESTAMP            : 01-DEC-12 15.58.16.106280 +00:00
PROXY_SESSIONID               :
GLOBAL_UID                    :
INSTANCE_NUMBER               : 0
OS_PROCESS                    : 16365
TRANSACTIONID                 : 08000B006E060000
SCN                           : 16684700
SQL_BIND                      :
SQL_TEXT                      :
OBJ_EDITION_NAME              :
------------------------------------------------

PL/SQL procedure successfully completed.

SQL>
```

We captured the audit turn off by auditing “**noaudit**”

At this point we would assume an attack so would need to detect this

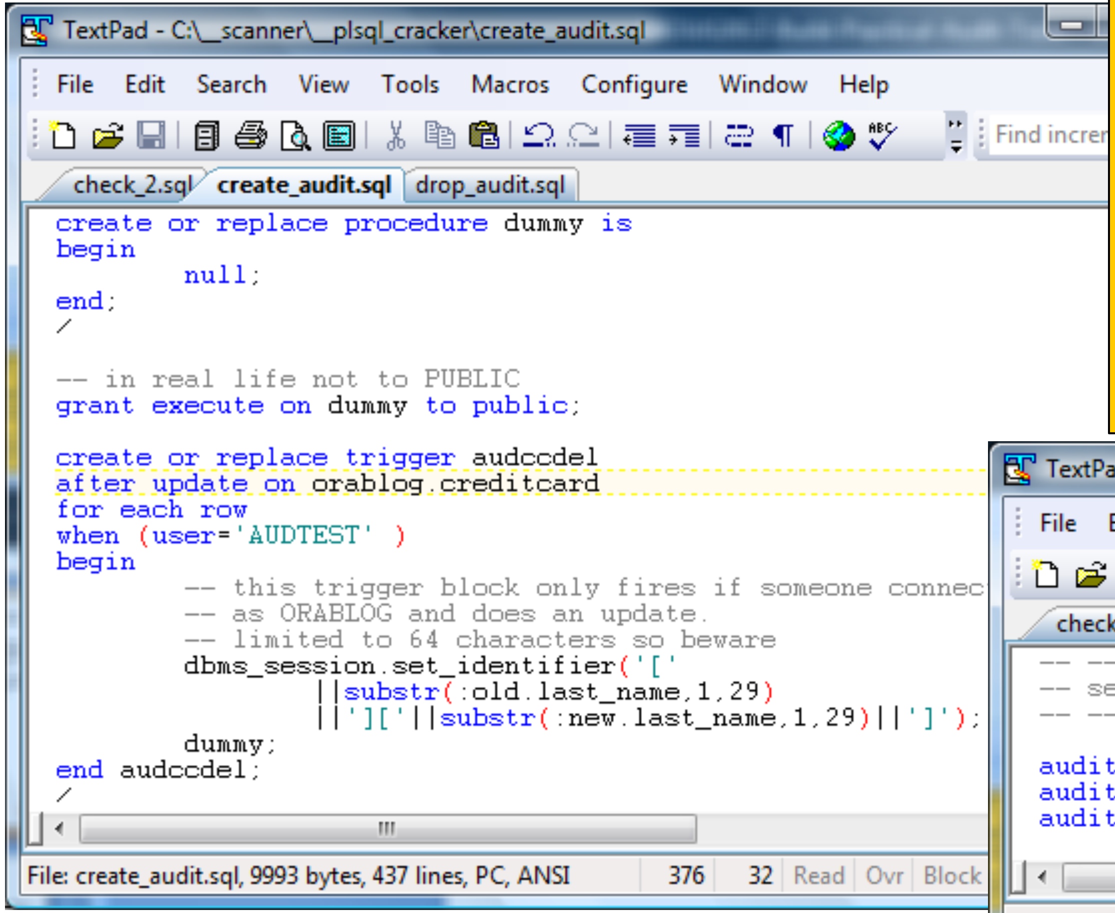
One way is to use the “dummy” trick which will capture the action

A good solution is to have multiple audit for correlation and also detect changes to audit

We can “poll” the settings and stability of the setup and also include a DDL trigger to prevent the change and also control privilege use.

We can also react by checking redo

Trick Set Up


TextPad - C:_scanner_plsql_cracker\create_audit.sql
File Edit Search View Tools Macros Configure Window Help
check_2.sql create_audit.sql drop_audit.sql

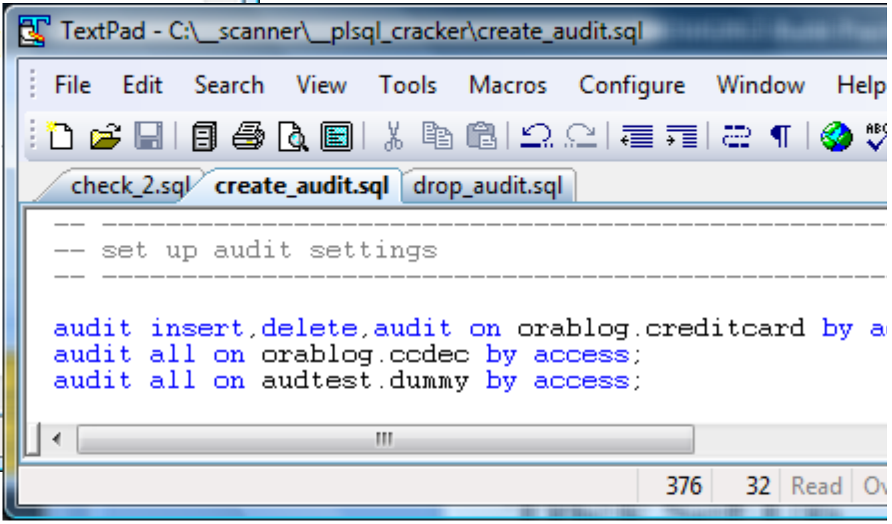
```
create or replace procedure dummy is
begin
        null;
end;
/

-- in real life not to PUBLIC
grant execute on dummy to public;

create or replace trigger audccdel
after update on orablog.creditcard
for each row
when (user='AUDTEST' )
begin
        -- this trigger block only fires if someone connec
        -- as ORABLOG and does an update.
        -- limited to 64 characters so beware
        dbms_session.set_identifier('['
                ||substr(:old.last_name,1,29)
                ||']['||substr(:new.last_name,1,29)||']');
        dummy;
end audccdel;
/
```

File: create_audit.sql, 9993 bytes, 437 lines, PC, ANSI 376 32 Read Ovr Block

Does one of 4 things:

1) Allow context based core audit
2) Allow performance improvements if written to OS
3) Allows correlation with second source
4) Can be used for before and after but very limited

No additional Database DML

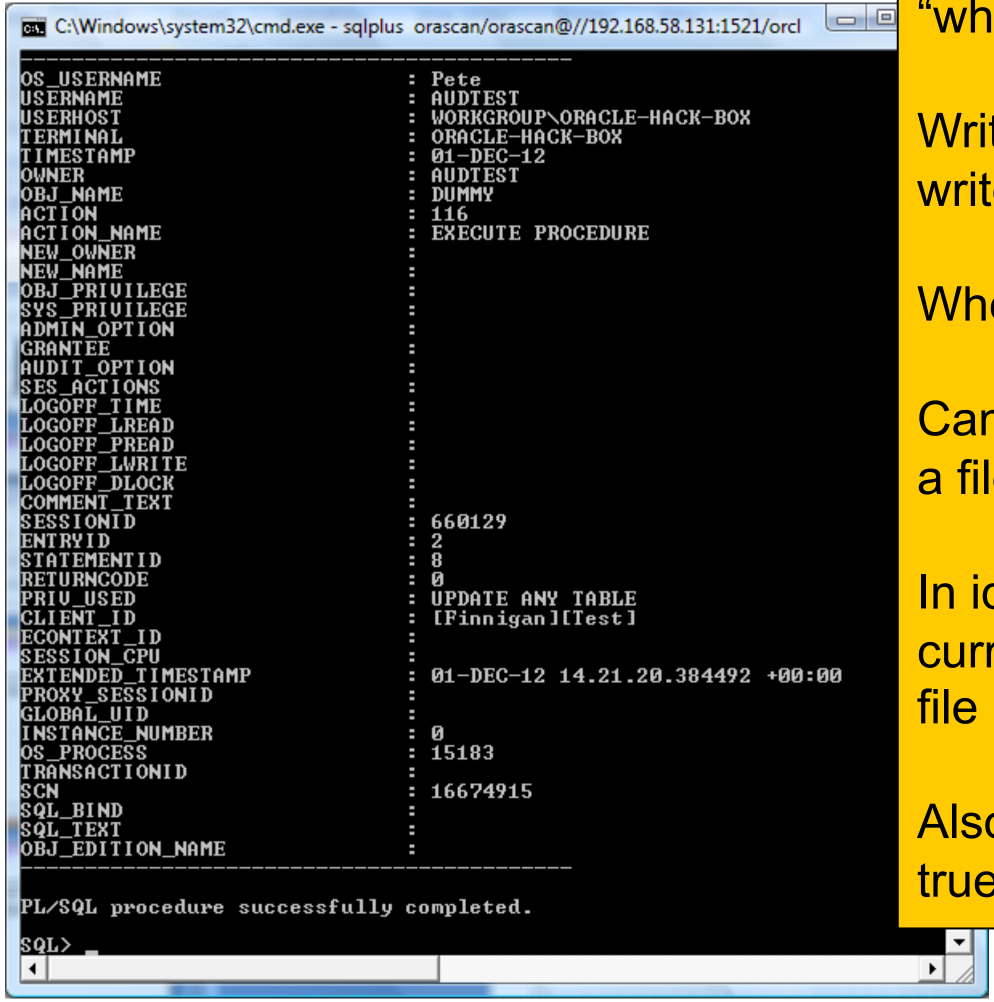

TextPad - C:_scanner_plsql_cracker\create_audit.sql
File Edit Search View Tools Macros Configure Window Help
check_2.sql create_audit.sql drop_audit.sql

```
-- -------------------------------------------------
-- set up audit settings
-- -------------------------------------------------

audit insert,delete,audit on orablog.creditcard by a
audit all on orablog.ccdec by access;
audit all on audtest.dummy by access;
```

376 32 Read O

PeteFinnigan.com Limited
Oracle Security

Trick 2


C:\Windows\system32\cmd.exe - sqlplus orascan/orascan@//192.168.58.131:1521/orcl

```
------------------------------------------------------------
OS_USERNAME                    : Pete
USERNAME                       : AUDTEST
USERHOST                       : WORKGROUP\ORACLE-HACK-BOX
TERMINAL                       : ORACLE-HACK-BOX
TIMESTAMP                      : 01-DEC-12
OWNER                          : AUDTEST
OBJ_NAME                       : DUMMY
ACTION                         : 116
ACTION_NAME                    : EXECUTE PROCEDURE
NEW_OWNER                      :
NEW_NAME                       :
OBJ_PRIVILEGE                  :
SYS_PRIVILEGE                  :
ADMIN_OPTION                   :
GRANTEE                        :
AUDIT_OPTION                   :
SES_ACTIONS                    :
LOGOFF_TIME                    :
LOGOFF_LREAD                   :
LOGOFF_PREAD                   :
LOGOFF_LWRITE                  :
LOGOFF_DLOCK                   :
COMMENT_TEXT                   :
SESSIONID                      : 660129
ENTRYID                        : 2
STATEMENTID                    : 8
RETURNCODE                     : 0
PRIV_USED                      : UPDATE ANY TABLE
CLIENT_ID                      : [Finnigan][Test]
ECONTEXT_ID                    :
SESSION_CPU                    :
EXTENDED_TIMESTAMP             : 01-DEC-12 14.21.20.384492 +00:00
PROXY_SESSIONID                :
GLOBAL_UID                     :
INSTANCE_NUMBER                : 0
OS_PROCESS                     : 15183
TRANSACTIONID                  :
SCN                            : 16674915
SQL_BIND                       :
SQL_TEXT                       :
OBJ_EDITION_NAME               :
------------------------------------------------------------

PL/SQL procedure successfully completed.

SQL> _
```

To improve performance we used a trick to create core audit from a trigger.

The trigger has little impact when the “when” clause is not TRUE

Writing to core audit saves database writes to a log table

When OS audit used best improvement

Can use utl_file and write before / after to a file and link file in client_id

In ideal world we need to retain the current “client id” for identity and add a file reference

Also the call to “dummy” needs to identify true source

Can use cron to poll, or dbms_scheduler. We can also get better performance with OS and SYSLOG audit but harder to write

Reporting And Alerts

- Starting with Audit in the database leads to simpler “to write” reports – we can use SQL
- Reports should be reports and alerts
  - Poll fast for changes to the security/audit
  - Poll fast for key issues and/or send alerts from triggers
  - Poll slower for summary of all checks
  - Overnight reports on issues located
  - Include sizing, management

Conclusions

- We must understand what is needed before deciding on a solution; people often decide on solutions first – this is wrong
- Practical for me means “risk” vs “cost” – i.e. Low Total cost
- Often risk can be mitigated with security on audit and audit on security and audit on audit
- Audit does not work fully unless identity is also included
- Audit does not work unless you can rely on the audit being not being changed – or at least you can detect the change and react

Questions?

Any Final Questions?

Building Practical Audit Trails

Design easy to implement and use audit trails