

# Adaptive Security in an Oracle Database

---



# Legal Notice

---

## Adaptive Security in an Oracle Database

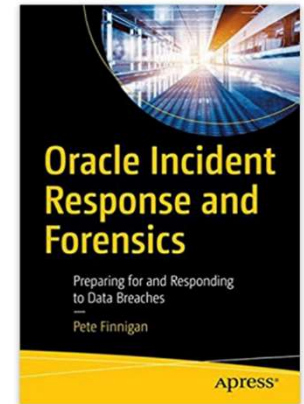
Published by  
PeteFinnigan.com Limited  
Tower Court  
3 Oakdale Road  
York  
England, YO30 4XL

Copyright © 2023 by PeteFinnigan.com Limited

No part of this publication may be stored in a retrieval system, reproduced or transmitted in any form by any means, electronic, mechanical, photocopying, scanning, recording, or otherwise except as permitted by local statutory law, without the prior written permission of the publisher. In particular this material may not be used to provide training of any type or method. This material may not be translated into any other language or used in any translated form to provide training. Requests for permission should be addressed to the above registered address of PeteFinnigan.com Limited in writing.

**Limit of Liability / Disclaimer of warranty.** This information contained in this course and this material is distributed on an “as-is” basis without warranty. Whilst every precaution has been taken in the preparation of this material, neither the author nor the publisher shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the instructions or guidance contained within this course.

**TradeMarks.** Many of the designations used by manufacturers and resellers to distinguish their products are claimed as trademarks. Linux is a trademark of Linus Torvalds, Oracle is a trademark of Oracle Corporation. All other trademarks are the property of their respective owners. All other product names or services identified throughout the course material are used in an editorial fashion only and for the benefit of such companies with no intention of infringement of the trademark. No such use, or the use of any trade name, is intended to convey endorsement or other affiliation with this course.



# Pete Finnigan – Background, 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 “Oracle Expert Practices”, “Oracle Incident Response and Forensics” books
- Oracle ACE for security
- Member of the OakTable
- Speaker at various conferences
  - UKOUG, PSOUG, BlackHat, more..
- Published many times, see
  - <http://www.petefinnigan.com> for links
- Influenced industry standards
  - And governments



# Agenda

---

- A bit of history
- Designing Audit Trails
- We use a toolkit
- Simple Policy based audit
- Adaptive Audit
- Adaptive Security



**PFCL**  
PETEFINNIGAN.COM LIMITED

## A bit of History

## History of the Audit Events, Toolkit and Talks

---

- In 2009 piece of work to help design audit trails
  - Site had limited staff, little time to design, deploy, maintain any audit trails
  - I came up with some simple ideas, proof of concepts – to package up audit trails for them; inc policy based audit, IPS and simple firewall
  - They spent limited time to deploy a useful audit trail
- Similar piece of work in 2011 where limited team needed to deploy audit
- 2012 to 2015 extended the toolkit
- I wrote a presentation back in 2012 and presented it just once at a SIG on practical audit trails where I mentioned this toolkit for the first time
- This then became the basis of a one day class on the same subject
- Reworked that presentation in UKOUG 2015 conference
- Customer in 2016 needed an audit trail to deploy quickly
- Deployed now to customers in UK, Ireland and Germany
- **The ideas of audit design came from these pieces of work and talks**

## Adaptive

---

- We must have audit to create adaptive audit
- We must have security to create adaptive security
- We most likely need audit to create adaptive security
- Audit is the key element



**PFCL**  
PETEFINNIGAN.COM LIMITED

# Designing Audit Trails



## Some People Think They Have Designed Audit

---

- I see sites with some audit settings
- This is not a WELL DESIGNED audit trail
- Usually these random set of parameters in the Oracle database will not catch a good range of events that could be an attack
- Some sites have application audit and OS audit
- BUT worse; lots of sites have no audit at the database engine level
- If there is a breach a lack of audit makes forensic response very difficult

## Design

---

- Before we get started implementing audit trails
- **Design** must be the first step
- The final chosen 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 – **we use our toolkit**
- Often people buy third party products and implement out of the box with no internal requirements!

Create audit events based on “I Want to know?”

## Create Audit Events

ID	Description	Category	Type	Report	Report Time
AE.1.0	Every connection to the database whether successful or not	ENGINE	COLLECT	NO	NONE
AE.1.1	Detect individuals sharing database one account	ENGINE	NORMAL	YES	SLOW
AE.1.2	Detect individuals who have access to multiple database accounts	ENGINE	NORMAL	YES	REGULAR
AE.1.3	Detect all failed logins	ENGINE	COLLECT	NO	NONE
AE.1.4	Detect a frequency of failed logins where the frequency is low (For example more than 3 per minute are detected)	ENGINE	NORMAL	YES	QUICK
AE.1.5	Detect a frequency of failed logins where the frequency is high (For example more than 50 per minute are detected). 1017, 28002 etc errors	SECURITY	ALERT	YES	IMMEDIATE
AE.1.6	Detect developer access ( <b>note: This will be allowed in development databases</b> )	ENGINE	NORMAL	YES	REGULAR
AE.1.7	Capture access to dormant accounts (3 months dormant)	ENGINE	NORMAL	YES	REGULAR
AE.2.0	Capture all DDL activity in the database	ENGINE	COLLECT	NO	NONE
AE.2.1	Capture structural changes (for instance tablespaces, data files)	ENGINE	NORMAL	YES	REGULAR
AE.2.2	Detect any user changes ( <b>legitimate</b> )	SECURITY	COLLECT	NO	NONE
AE.2.3	Detect any user changes ( <b>not legitimate</b> )	SECURITY	ALERT	YES	IMMEDIATE
AE.2.4	Detect profile changes	SECURITY	NORMAL	YES	QUICK
AE.2.5	Detect any GRANTS for roles, system privileges or objects ( <b>not legitimate</b> )	SECURITY	ALERT	YES	IMMEDIATE

## Build On The Audit Events

---

- Work backwards from the events to decide what raw audit to collect
- Then how to work out if event has occurred
- Then how to report
- Then how to alert
- Then how to escalate
- When you have this “table” decide on the technical solution that can be implemented and deployed

# Audit Trail Toolkit

## The Goal of the Toolkit

---

- As simple as `SQL> @atk` and a sophisticated audit trail is up and running
- Making it simple for organisations to deploy audit trails simply, with no resources
- No design, implement, test etc as we have done it for you already
- Used in ATC mode – space also is managed in each target database audited
- Simple to configure or not configured at all
- A complete solution to know what is happening at the database engine level for sites with limited resources

## PFCLATK – “A”udit “T”rail tool”K”it

---

- Toolkit to aid audit trail deployment easily
- Simple pre-configure
- Policy based
- Alert based
- Multiple audit trails sources
- Add in factors (input hints)
- Separated schema design
- Manual 27 pages currently
- Version 2.4.0.0 currently
- Layered audit

Can be deployed to multiple databases and a central database

## Elements of Design - Core

---

- The core features of our audit design are
  - Policies and events (There are 20 policies and 17 events)
  - Polled jobs and reports as checks (There are 7 job intervals)
  - Core PL/SQL API Package
  - Rules/policy/jobs/payload meta data
  - Audit trails specific to PFCLATK
  - System Triggers
  - DDL triggers
  - DML triggers
  - PL/SQL based
- Checks can be made via polled jobs to test if the audit is valid
- Raw data is collected, filters are jobs polled via DBMS\_SCHEDULER; results added to alerts and alerts generates escalation





## Factors

Some factors are re-defined, some should be edited and more can be added easily

Factors allow the toolkit to be customised for a specific site

```
100 -----
101 atk.pfclatk.addfactor('SUPPORT-IP', '192.168.56.1');
102 -----
103 -- LOW-FAILED: Define the number of failed logins per minute above
104 --                which an alert should be raised. This should be specified
105 --                per 30 minutes. So a number of 3 per minute would be set
106 --                to 90 for the 30 minute period.
107 -----
108 atk.pfclatk.addfactor('LOW-FAILED', '60');
109 -----
110 -- HIGH-FAILED: Define the number of failed logins above which an alert
111 --                would be raised. This should be specified per 30 minutes
112 --                So for a number of 50 per minute that would be 30*50 =
113 --                1500. This could indicate a scripted attack.
114 -----
115 atk.pfclatk.addfactor('HIGH-FAILED', '1500');
116 -----
117 -- DEV-IP: Define the IP address of all of your developers terminals
118 --                here.
119 --
120 -- NOTE: For multiple developers add multiple entries here.
121 -----
122 atk.pfclatk.addfactor('DEV-IP', '192.168.56.1');
123 -----
124 -- DBA-USER: Define the DBA user accounts allowed to be used as DBA.
125 --                This could be SYS and SYSTEM but should really be separate
126 --                user accounts for each DBA
127 --
128 -- NOTE: For multiple DBA add multiple entries here.
129 -----
130 atk.pfclatk.addfactor('DBA-USER', 'SYS');
131 atk.pfclatk.addfactor('DBA-USER', 'SYSTEM');
132 -----
133 -- ERROR-RATE: This is the trigger rate for the number of errors that
134 --                can occur for a single user/IP that could indicate an
135 --                attack.
136 -----
137 atk.pfclatk.addfactor('ERROR-LIMIT', '8');
138 --
139 end;
140 /
...
```



# Audit Policies

---

```
270 begin
271 -- create the policy
272 atk.pfclatk.createpolicy('PROFILEPRIVILEGE');
273
274 -- audit profiles
275 atk.pfclatk.createandaddrule('PROFILEPRIVILEGE','Create Profile','CORE-S','create profile');
276 atk.pfclatk.createandaddrule('PROFILEPRIVILEGE','Drop Profile','CORE-S','drop profile');
277 atk.pfclatk.createandaddrule('PROFILEPRIVILEGE','Alter Profile','CORE-S','alter profile');
278 atk.pfclatk.createandaddrule('PROFILEPRIVILEGE','Profile','CORE-S','profile');
279
280 -----
281 -- Add filter job to detect non-legitimate profile changes - i.e. use
282 -- of PROFILE system privileges; not use of statement PROFILE and not use
283 -- of a DBA IP address and not use of a DBA account; so if a DBA
284 -- uses a non DBA account from his own IP it should be detected
285 -----
286
287 atk.pfclatk.addfilter('NON-AUTH-PROFILE-CHANGE','PROFILEPRIVILEGE','HALF HOUR',
288     '[Alert] Non-legitimate profile privilege change',
289     'select ''A non-authorized user change {'||a.action_name||''} on {'||a.obj_name||''} by
290
291 -- enable the policy
292 atk.pfclatk.enablepolicy('PROFILEPRIVILEGE');
293 end;
294 /
295
```

- Policies declare collection of raw data and also events
- PFCLATK policies are different to Unified audit – we filter on collected data after storage to look for abuse; Unified audit filters before storage
- Core audit, DML, System triggers

## Audit of Audit

---

- A multi-layer approach is needed
  - Audit of core trail tables such as AUD\$
  - Audit of core audit settings such as AUDIT\$
  - Audit of triggers (Event, DDL and DML)
  - Audit of custom logs
  - Audit of audit functionality, packages and other objects
  - All can be set up as policies in PFCLATK



## Simple Policy Based Audit Trails

# Policies I Will Implement

```
Command Prompt - sqlplus /nolog
SQL> select * from atkd.pfclatk_policy;
 1 CONNECT                                Y
 2 USERPRIVILEGE                          Y
 3 PROFILEPRIVILEGE                       Y
 4 ERROR                                  Y
 5 AUDITAUDIT                             Y
 6 BOUNCE                                  Y
 7 AUDITSEC                               Y
 8 AUDITSECONAUDIT                        Y
 9 METADATA                               Y
10 EXTERNALS                              Y
11 DANGEROUS                              Y
12 EXTERNALSDDL                           Y
13 PARAMETERS                             Y
14 SYSTEM                                  Y
15 STRUCTURAL                             Y
16 DDL                                     Y
17 ALLSTATEMENTS                          N
18 SYSROLES                               N
19 ALLPRIV                                 N
20 ALL                                     N
21 SECCONF                                Y
22 SCHEMAOBJ                              Y

22 rows selected.

SQL>
```

# Install the Audit Events and Toolkit

```
Command Prompt - sqlplus /nolog

PFCLATK: Release 2.2.0.0 - Production on Wed Sep 22 14:45:38 2021
Copyright (c) 2009 - 2019 PeteFinnigan.com Limited. All rights reserved.

SECTION-[1] - Remove existing schemas and users
SECTION-[2] - Create the Schema owner ATK (Functional owner)
    [2-1] Create ATK Schema
    [2-2] Perform ATK Grants
SECTION-[3] - Create schema owner ATKD (Data owner)
    [3-1] Create ATKD schema
    [3-2] Perform ATKD Grants
SECTION-[4] - Create PFCLAudit Roles
    [4-1] Create ATK_ADMIN Role
    [4-2] Create ATK_REPORT Role
    [4-3] Revoke roles from SYS
SECTION-[5] - Connect to ATKD and Create Objects
    [5-1] Create AUD$ for PUL To Extract Data
    [5-2] Perform grants on ATKD.AUD$
    [5-3] Create the main info table
    [5-4] Perform grants on the info table
    [5-5] Perform grant in info table to the admin role
    [5-6] Create the Info Data
    [5-7] Create version history table
    [5-8] Create the policy sequence
    [5-9] Grant permissions on the table
    [5-10] Grant permissions on the sequence
    [5-11] Add version history
    [5-12] Create the Rules table
    [5-13] Create the rules sequence
    [5-14] Perform grants on rules table
```



# Some Audit Results From Hacks

Warning: oci\_execute() [function.oci-execute]: ORA-00907: missing right parenthesis in /usr/local/apache2/htdocs/wp-includes

Orablog database error: []

```
SELECT * FROM (SELECT a.*, rownum RN FROM ( SELECT * FROM wp_posts WHERE 1=1 AND (((post_title LIKE '%x%'))--%) OR (post_content LIKE '%x%'))--%) OR (post_title LIKE '%x%'))--%) OR (post_date_gmt <= SYSDATE AND (post_status = 'publish') AND post_status != 'attachment' ORDER BY post_date DESC ) a WHERE rownum <= 10) WHERE rn >= 1
```

Command Prompt - sqlplus /nolog

```
Press any key to continue....

Connected.
USER is "DEV"
ERROR:
ORA-01756: quoted string not properly terminated

BEGIN orablog.custa('x' union select username from dba
*
ERROR at line 1:
ORA-00942: table or view does not exist
ORA-06512: at "ORABLOG.CUSTA", line 9
ORA-06512: at line 1

BEGIN orablog.custa('x' union select username,created
*
ERROR at line 1:
ORA-01789: query block has incorrect number of result c
ORA-06512: at "ORABLOG.CUSTA", line 9
ORA-06512: at line 1
```

Command Prompt - sqlplus /nolog

```
DOC>
DOC>| =====
DOC>| lets change the orablog users password!
DOC>| =====
DOC>|
DOC>| SQL> connect dev/dev@//192.168.56.86:1521/pdborcl.localdomain
DOC>| SQL> sho user
DOC>| SQL> set serveroutput on
DOC>|
DOC>| SQL> create or replace function xx
DOC>| SQL> return varchar2
DOC>| SQL> authid current_user as
DOC>| SQL> pragma autonomous_transaction;
DOC>| SQL> begin
DOC>| SQL> execute immediate 'alter user orablog identified by password';
DOC>| SQL> return 'xx';
DOC>| SQL> end;
DOC>| SQL> /
DOC>|
DOC>| SQL> sho err
DOC>|
DOC>| SQL> grant execute on xx to public;
DOC>|
DOC>| SQL> exec orablog.custa('x' union select dev.xx from dual--');
DOC>|
DOC>
DOC>#
```

pages  
about  
contact page

blogroll  
oracle security expertise  
pfclscan

categories:  
uncategorized  
uncategorized

# Audit Trail Report

- We catch the NOAUDIT
- We catch the password changes
- We catch all errors, 942, 1789, 907, 911 etc
- We catch the creation of the hack procedure
- **All events are not caught 100% but the attack was not based on the audit design!!**

```

Command Prompt - sqlplus /nolog

DD-BE 22-SEP-21 03.10.52.713992 PM +01:00 DEV      grant execute on xx          DEV      WORKGROUP\DESKTOP-R  Pete      163002 192.168.56.1
AUDIT 22-SEP-21 03.10.52.716622 PM +01:00 DEV      DEV.XX                       GRANT OBJECT                WORKGROUP\DESKTOP-R7 Pete      163002
AUDIT 22-SEP-21 03.10.52.728008 PM +01:00 DEV      ORABLOG.CUSTA               EXECUTE PROCEDURE          WORKGROUP\DESKTOP-R7 Pete      163002
AUDIT 22-SEP-21 03.10.52.729926 PM +01:00 DEV      ORABLOG.CREDIT_CARD        SESSION REC                 WORKGROUP\DESKTOP-R7 Pete      163002
NO-BE 22-SEP-21 03.10.52.732449 PM +01:00 DEV      noaudit select on o        DEV      WORKGROUP\DESKTOP-R  Pete      163002 192.168.56.1
DD-BE 22-SEP-21 03.10.52.733946 PM +01:00 DEV      noaudit select on o        DEV      WORKGROUP\DESKTOP-R  Pete      163002 192.168.56.1
AUDIT 22-SEP-21 03.10.52.736187 PM +01:00 DEV      ORABLOG.CREDIT_CARD        NOAUDIT OBJECT             WORKGROUP\DESKTOP-R7 Pete      163002
LOGOF 22-SEP-21 03.10.54.609230 PM +01:00 DEV      .                            LOGOFF                      DEV      WORKGROUP\DESKTOP-R  Pete      163002 192.168.56.1
AUDIT 22-SEP-21 03.10.54.612925 PM +01:00 DEV      .                            LOGOFF                      WORKGROUP\DESKTOP-R7 Pete      163002
LOGON 22-SEP-21 03.10.54.661608 PM +01:00 SYS      .                            LOGON                       SYS      WORKGROUP\DESKTOP-R  Pete      4294967295 192.168.56.1
LOGOF 22-SEP-21 03.11.00.140215 PM +01:00 SYS      .                            LOGON                       SYS      WORKGROUP\DESKTOP-R  Pete      4294967295 192.168.56.1
AUDIT 22-SEP-21 03.11.00.196948 PM +01:00 DEV      .                            LOGON                       WORKGROUP\DESKTOP-R7 Pete      163003 Authenticated b CREATE S
SESSION

y: DATABASE; Cl
ient address: (
ADDRESS=(PROTC
OL=tcp)(HOST=19
2.168.56.1)(POR
T=1043))

LOGON 22-SEP-21 03.11.00.198875 PM +01:00 DEV      .                            LOGON                       DEV      WORKGROUP\DESKTOP-R  Pete      163003 192.168.56.1
AUDIT 22-SEP-21 03.11.00.200549 PM +01:00 DEV      SYS.DBMS_DEBUG_JDWP        EXECUTE PROCEDURE          WORKGROUP\DESKTOP-R7 Pete      163003
AUDIT 22-SEP-21 03.11.00.201956 PM +01:00 DEV      SYS.AUD$                   SELECT                      WORKGROUP\DESKTOP-R7 Pete      163003
DD-BE 22-SEP-21 03.11.00.213281 PM +01:00 DEV      create or replace f        DEV      WORKGROUP\DESKTOP-R  Pete      163003 192.168.56.1
AUDIT 22-SEP-21 03.11.00.222751 PM +01:00 DEV      DEV.XX                     CREATE FUNCTION             WORKGROUP\DESKTOP-R7 Pete      163003          CREATE P
ROCEDU

RE

DD-BE 22-SEP-21 03.11.00.241730 PM +01:00 DEV      grant execute on xx          DEV      WORKGROUP\DESKTOP-R  Pete      163003 192.168.56.1
AUDIT 22-SEP-21 03.11.00.242732 PM +01:00 DEV      SYS.DBMS_DEBUG_JDWP        EXECUTE PROCEDURE          WORKGROUP\DESKTOP-R7 Pete      163003
AUDIT 22-SEP-21 03.11.00.245078 PM +01:00 DEV      DEV.XX                       GRANT OBJECT                WORKGROUP\DESKTOP-R7 Pete      163003
AUDIT 22-SEP-21 03.11.00.255571 PM +01:00 DEV      ORABLOG.CUSTA               EXECUTE PROCEDURE          WORKGROUP\DESKTOP-R7 Pete      163003
DD-BE 22-SEP-21 03.11.00.260781 PM +01:00 DEV      alter user orablog         DEV      WORKGROUP\DESKTOP-R  Pete      163003 192.168.56.1
AUDIT 22-SEP-21 03.11.00.264281 PM +01:00 DEV      .ORABLOG                   ALTER USER                  WORKGROUP\DESKTOP-R7 Pete      163003
LOGOF 22-SEP-21 03.11.02.915059 PM +01:00 DEV      .                            LOGOFF                      DEV      WORKGROUP\DESKTOP-R  Pete      163003 192.168.56.1
AUDIT 22-SEP-21 03.11.02.918414 PM +01:00 DEV      .                            LOGOFF                      WORKGROUP\DESKTOP-R7 Pete      163003
LOGON 22-SEP-21 03.11.02.969985 PM +01:00 SYS      .                            LOGON                       SYS      WORKGROUP\DESKTOP-R  Pete      4294967295 192.168.56.1
AU-BE 22-SEP-21 03.11.02.986466 PM +01:00 SYS      audit select on ora        SYS      WORKGROUP\DESKTOP-R  Pete      4294967295 192.168.56.1
DD-BE 22-SEP-21 03.11.02.988030 PM +01:00 SYS      audit select on ora        SYS      WORKGROUP\DESKTOP-R  Pete      4294967295 192.168.56.1
DD-BE 22-SEP-21 03.11.02.995716 PM +01:00 SYS      alter user orablog         SYS      WORKGROUP\DESKTOP-R  Pete      4294967295 192.168.56.1
LOGON 22-SEP-21 03.11.14.072448 PM +01:00 SYS      .                            LOGON                       SYS      oel1124.localdomain oracle 163004

```





**PFCL**  
PETEFINNIGAN.COM LIMITED

# Adaptive Audit



# Adaptive Audit

---

- **Change the audit trail based on security at the time**
- Based on a real attack happening – in progress – or a perceived attack happening the audit capture can be increased
- This, as with adaptive security would be reduced after the perceived attack
- Adaptive Audit is useful because in normal circumstances we would not want to collect excessive audit trails but during an attack we may do so.
- For example if the application schema has limited audit normally we increase to audit all access during an attack
- These ideas need to be designed carefully to ensure that the audit does not become a "Denial Of Service"
- Example
  - A schema changes its password from the web server; so audit all actions in the session or audit all actions in the schema by all users

# What is Conditional Audit?

---

- Audit would be more powerful if conditionally applied
  - Oracles Fine Grained Audit (FGA) attests to this
  - Oracles Unified Audit from 12c also attests to this but still is underpinned by core audit facilities
- Trigger based security can use some element of conditionality
  - When clause can be used to limit the trigger body to certain conditions such as time based, user based privilege based and SYS\_CONTEXT() based and more
  - OF clause can limit a trigger to certain columns BUT has little value in conditional audit other than protection – see integrity
  - A trigger body is PL/SQL so any conditional audit can be programmed in
- Secure contexts can be used to control granularity in triggers, audit functions and more across the database – set up at logon
- Conditional security in core audit is not feasible BUT
  - We can process the records records after collection via reports/jobs
  - This is in essence what unified audit is doing



# Adaptive Security

No one is doing this BUT WE COULD

The ultimate – one database is being attacked and it “tells” all others to change audit and security

# Adaptive Security

---

- Adaptive security is not part of audit trail design but adaptive security could be controlled by audit results and reports
- Create multiple levels of security i.e. defcon5, 4, 3, 2,1
- Adaptive security is where
  - Security is changed dynamically based on a real attack happening or perceived real attack happening
  - Normally this would be to tighten security to defend the database because of the current attack
  - After an attack the reverse could normally occur to remove the additional restrictions
- Examples
  - Detect change of a password and lock the user whose password was changed
  - Detect change of audit by a schema (user) and kill the user doing it
  - These can be implemented in triggers as VP/IPS or specific payloads as jobs

## Conclusions

---

- Secure the data in your database
- Include an audit trail
- When you are secure
  - Think about adaptive audit
  - Think about adaptive security

# Questions

---

?

# Adaptive Security in an Oracle Database

---