WHO SHUT MY DATABASE DOWN??

accentureoperations

FRITS HOOGLAND

$(whoami)

Frits Hoogland

Working with Oracle products since 1996

Blog: http://fritshoogland.wordpress.com

Twitter: @fritshoogland

Email: frits.hoogland@enkitec.com

Oracle ACE Director

OakTable Member

VERSIONS

- **Versions used in this presentation:**
  - **Database: Oracle version 12.1.0.2**
  - **Operating system: Oracle Linux 6.8**

ABOUT AUDITING

- Dirty word!

- In most companies auditing is either:
  - Not implemented/left untouched.
  - Implemented as part of security rules.

- In this investigation I use default settings of database and linux.
  - Most of the information in this presentation is usable for ‘post mortem’ investigation.

DATABASE: MANDATORY AUDITING

- Regardless of database audit settings, all databases perform mandatory auditing.
  - This audit information is written to .aud files*.
  - The location of these files is set with the AUDIT_FILE_DEST parameter.
    - Default: $ORACLE_HOME/rdbms/audit.
  - This applies to the ASM instance too.
- Audit files are owned by the user under which the database runs.
  - This is probably the same user which is stopping and starting a database.
  - This means they can be tampered with.
  - But it's the only way to get information on startup or shutdown IN A DEFAULT CONFIG.

DATABASE: MANDATORY AUDITING

- This is how a database audit record looks like:

```
Mon Jan 30 12:53:59 2017 +00:00
LENGTH : '161'
ACTION :[18] 'SHUTDOWN IMMEDIATE'
DATABASE USER:[1] '/'
PRIVILEGE :[6] 'SYSDBA'
CLIENT USER:[6] 'oracle'
CLIENT TERMINAL:[5] 'pts/0'
STATUS:[1] '0'
DBID:[0] ‘'
```

- This means that the ONLY thing that describes something unique about the session that stops (or starts) a database is the terminal name!

DATABASE: MANDATORY AUDITING

- Having audit records in text files that span multiple lines make it challenging to use CLI tools for investigation.

- If you need to keep track of multiple logfiles and want to create fields of certain values, a log gathering product like the 'ELK stack' or splunk is very convenient.
  - Interested? See: https://fritshoogland.wordpress.com/2017/01/21/auditing-oracle-database-stopping-and-starting-using-the-elk-stack/
  - Essentially:
    - Filebeat runs on the local machines, monitoring logfiles and shipping log entries.
    - Logstash acts as a collector and enrichment hub.
    - Elasticsearch is the data store, which has full-text search capabilities.

DATABASE: MANDATORY AUDITING - FILEBEAT

- Log files for which a single line does not need to joined back to other lines:

```
- input_type: log
  paths:
    - /var/log/messages
  document_type: messages
```

- Oracle audit files, multiline:

```
- input_type: log
  paths:
    - /u01/app/oracle/admin/*/adump/*.aud
  document_type: oracle_audit
  multiline:
    pattern: '^[A-Za-z]{3} [A-Za-z]{3} [0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2} [0-9]{4
    negate: true
    match: after
```

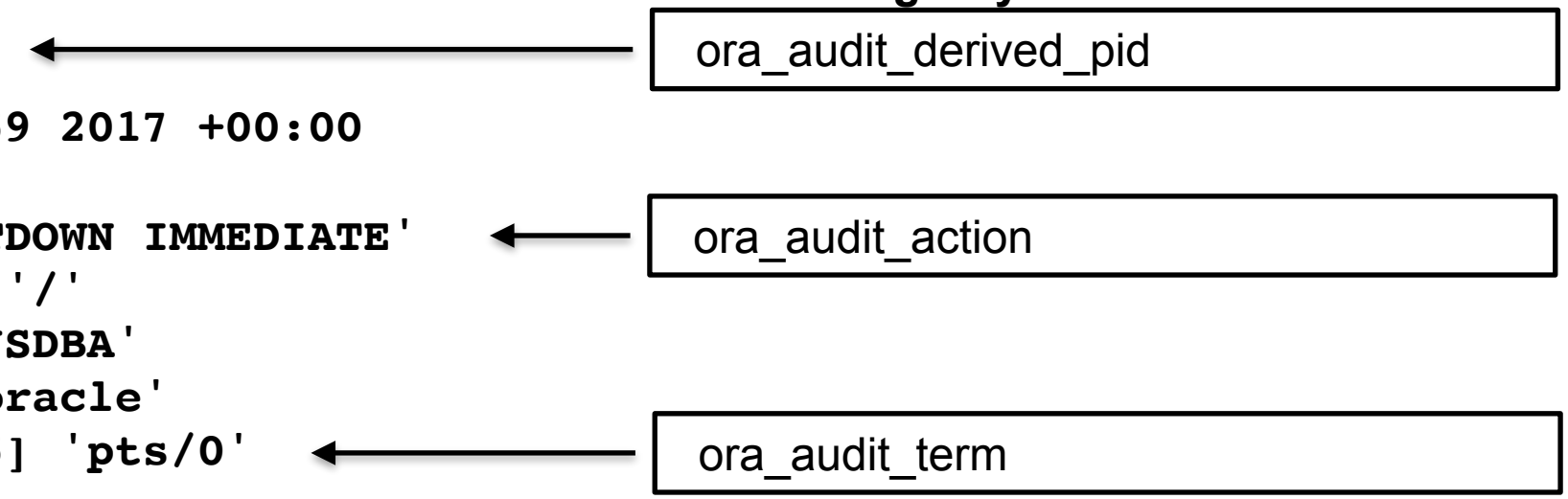

DATABASE: MANDATORY AUDITING - LOGSTASH

```
filter {
  if [type] == "oracle_audit" {
    grok {
      match => { "message" => "^%{DAY} %{MONTH:M} %{MONTHDAY:d} %{HOUR:h}:%{MINUTE:m}:%{SECOND:s} %{YEAR:y}" }
      add_tag => [ "grok", "oracle_audit" ]
    }
    grok {
      match => { "message" => "ACTION :\[[0-9]*\] '(?<ora_audit_action>.*)'.*DATABASE USER:\[[0-9]*\] '(?
<ora_audit_dbuser>.*)'.*PRIVILEGE :\[[0-9]*\] '(?<ora_audit_priv>.*)'.*CLIENT USER:\[[0-9]*\] '(?
<ora_audit_osuser>.*)'.*CLIENT TERMINAL:\[[0-9]*\] '(?<ora_audit_term>.*)'.*STATUS:\[[0-9]*\] '(?
<ora_audit_status>.*)'.*DBID:\[[0-9]*\] '(?<ora_audit_dbid>.*)'"}
    }
    grok {
      match => { "source" => [ ".*/[a-zA-Z0-9_#$]*_[a-z0-9]*_(?<ora_audit_derived_pid>[0-9]*)_[0-9]*\.aud" ] }
    }
    mutate {
      add_field => { "ts" => "%{y}-%{M}-%{d} %{h}:%{m}:%{s}" }
    }
    date {
      locale => "en"
      match => [ "ts", "YYYY-MMM-dd HH:mm:ss" ]
    }
    mutate {
      remove_field => [ "ts", "y", "M", "d", "h", "m", "s" ]
    }
  }
}
```

accenture operations

DATABASE: MANDATORY AUDITING

- I configured logstash to create fields in Elasticsearch in the following way:



Time	ora_audit_action	ora_audit_term	ora_audit_derived_pid
January 30th 2017, 19:31:08.000	SHUTDOWN IMMEDIATE	pts/1	18246
January 30th 2017, 19:31:08.000	ALTER DATABASE CLOSE NORMAL	pts/1	18246
January 30th 2017, 19:31:08.000	ALTER DATABASE DISMOUNT	pts/1	18246
January 30th 2017, 19:31:03.000	SELECT SYS_CONTEXT('USERENV','CDB_NAME'), SYS_CONTEXT('USERENV','CON_NAME') FROM SYS.DUAL	pts/1	18246
January 30th 2017, 19:30:46.000	CONNECT	pts/1	18246
January 30th 2017, 19:30:46.000	COMMIT	pts/1	18246
January 30th 2017, 19:30:46.000	COMMIT	pts/1	18246

Time	ora_audit_action	ora_audit_term	ora_audit_derived_pid
January 30th 2017, 17:37:26.000	ALTER DATABASE OPEN	pts/1	9683
January 30th 2017, 17:37:25.000	ALTER DATABASE MOUNT	pts/1	9660
January 30th 2017, 17:37:25.000	CONNECT	pts/1	9683
January 30th 2017, 17:37:23.000	ALTER SYSTEM REGISTER	-	9659
January 30th 2017, 17:37:18.000	SELECT value FROM vSparameter WHERE name = :1	-	9659
January 30th 2017, 17:37:18.000	ALTER SESSION SET EVENTS '10165 TRACE NAME CONTEXT FOREVER'	-	9659
January 30th 2017, 17:37:18.000	ALTER SYSTEM SET LOCAL_LISTENER=' (ADDRESS= (PROTOCOL=TCP)(HOST=10.0.3.11)(PORT=1521))' SCOPE=MEMORY SID='oohg' /* db agent *//* {0:0:183} */	-	9659
January 30th 2017, 17:37:18.000	STARTUP	-	9568
January 30th 2017, 17:37:18.000	SELECT value FROM vSparameter WHERE name = :1	-	9659
January 30th 2017, 17:37:18.000	select cdb from v$database	-	9659
January 30th 2017, 17:37:18.000	SELECT DECODE(null,'','Total System Global Area','') NAME_COL_PLUS_SHOW_SGA, SUM(VALUE), DECODE (null,'', 'bytes','') units_col_plus_show_sga FROM V$SGA UNION ALL SELECT NAME NAME_COL_PLUS_SHOW_SGA , VALUE, DECODE (null,'', 'bytes','') units_col_plus_show_sga FROM V$SGA	pts/1	9660
January 30th 2017, 17:37:18.000	ALTER SESSION SET "_notify_crs" = false	-	9659
January 30th 2017, 17:37:18.000	CONNECT	pts/1	9660
January 30th 2017, 17:37:18.000	CONNECT	-	9659
January 30th 2017, 17:37:18.000	SELECT value FROM vSparameter WHERE name = :1	-	9659
January 30th 2017, 17:37:12.000	CONNECT	pts/1	9568

LINUX AUDIT DATA

- In order to understand who manipulated a database, we need to relate the database audit data with linux audit data.
  - To be concrete: we need to relate the (database) terminal information with a (linux) session.

- A lot of linux utilities use syslog to log information. All security/authorization related messages are written to /var/log/secure and use the authpriv 'facility':
  - /etc/rsyslog.conf: authpriv.* /var/log/secure

```
Jan 30 13:05:57 frits-target8 sshd[71237]: Accepted publickey for opc
from 217.63.231.46 port 38428 ssh2
Jan 30 13:00:51 frits-target8 su: pam_unix(su-l:session): session opened
for user root by opc(uid=0)
Jan 30 13:00:51 frits-target8 sudo:      opc : TTY=pts/0 ; PWD=/home/
opc ; USER=root ; COMMAND=/bin/su -
```


LINUX AUDIT DATA

- Additional to /var/log/secure:
  - ‘last’ command: successful logins, if the file /var/log/wtmp exists.
  - ‘lastb’ command: unsuccessful logins, if the file /var/log/btmp exists.

- On a default system, /var/log/wtmp exists, /var/log/btmp doesn’t.
- The /var/log/*tmp files are not in readable format, the last utility needs to be used.
- Output:

```
# last | less
oracle   pts/1        192.168.66.1     Mon Jan 30 13:17   still logged in
oracle   pts/1        192.168.66.1     Mon Jan 30 13:16 - 13:17  (00:00)
vagrant  pts/0        10.0.2.2         Mon Jan 30 13:14   still logged in
```


LINUX AUDIT DATA

- When no sudo is used, and logins are directly done as the oracle software owner, the last utility can provide all available details:

```
Mon Jan 30 19:53:19 2017 +00:00
LENGTH : '161'
ACTION :[18] 'SHUTDOWN IMMEDIATE'
DATABASE USER:[1] '/'
PRIVILEGE :[6] 'SYSDBA'
CLIENT USER:[6] 'oracle'
CLIENT TERMINAL:[5] 'pts/1'
STATUS:[1] '0'
DBID:[0] ''

# last -t 20170130195319 pts/1 -n 1
oracle   pts/1        192.168.66.1     Mon Jan 30 15:20    gone - no logout
```


LINUX AUDIT DATA

- When sudo is used, the last utility provides the logon session:

```
Mon Jan 30 21:05:45 2017 +00:00
LENGTH : '161'
ACTION :[18] 'SHUTDOWN IMMEDIATE'
DATABASE USER:[1] '/'
PRIVILEGE :[6] 'SYSDBA'
CLIENT USER:[6] 'oracle'
CLIENT TERMINAL:[5] 'pts/1'
STATUS:[1] '0'
DBID:[0] ‘'

# last -t 20170130210545 pts/1 -n 1
opc      pts/1        316.93.231.22    Mon Jan 30 21:04   still logged in
```


LINUX AUDIT DATA

- But you have to go through the /var/log/secure file to learn about the invocation of sudo:

```
# grep sudo /var/log/secure
Jan 30 21:05:00 frits-target8 sudo:      opc : TTY=pts/1 ; PWD=/home/opc ;
USER=root ; COMMAND=/bin/su - oracle
```

- Careful align the sudo and wtmp times.

```
# last -t 20170130210545 pts/1 -n 1
opc      pts/1        316.93.231.22    Mon Jan 30 21:04   still logged in
```


LINUX AUDIT DATA

- **Okay! Problem solved!**
- **Would this be the shortest presentation ever?**
- **Well…not really…have a look at the following oracle audit data:**

Time	ora_audit_action	ora_audit_term	ora_audit_derived_pid
January 30th 2017, 22:27:34.000	SHUTDOWN IMMEDIATE	-	32156
January 30th 2017, 22:27:34.000	ALTER DATABASE CLOSE NORMAL /* db agent *//* {0:0:298} */	-	32156
January 30th 2017, 22:27:34.000	ALTER DATABASE DISMOUNT /* db agent *//* {0:0:298} */	-	32156
January 30th 2017, 22:27:29.000	ALTER SESSION SET "_notify_crs" = true	-	32156
January 30th 2017, 22:27:29.000	CONNECT	-	32156
January 30th 2017, 22:27:29.000	select cdb from v$database	-	32156
January 30th 2017, 22:27:29.000	ALTER SESSION SET EVENTS '10165 TRACE NAME CONTEXT FOREVER'	-	32156

ORACLE CLUSTERWARE

- Whenever clusterware is used, and:
  - Autostart of clusterware is enabled, or
  - The server is stopped on the operating system level (poweroff/shutdown commands), or
  - Either ‘crsctl’ or ‘srvctl’ is used to stop or start a database or instance.

- The database state is changed by a clusterware process!
  - This renders the database audit records useless for identifying the session that initiated stopping or starting.

- Clusterware does not provide client session auditing facilities.

WHEN CLUSTERWARE IS USED WITH THE DATABASE,

AND CRSCTL OR SRVCTL IS USED TO MANIPULATE THE DATABASE STATE,

IT'S IMPOSSIBLE TO LINK THE DATABASE STATE CHANGE TO AN OPERATING SYSTEM SESSION.

LINUX KERNEL AUDIT FRAMEWORK

- After searching for audit functionality beyond last & syslog I found the linux kernel audit framework.
  - In-kernel, always-on, audit functionality!
  - The audit framework is in the kernel for quite some time (since RHEL/OL 4).
  - Reasonably unknown.

- By default login and logoff related events are audited.
  - Includes cryptology, authentication, accounting and session related information.
  - Quite verbose.

LINUX AUDIT DATA - EXAMPLE

```
type=CRYPTO_KEY_USER msg=audit(1485857482.532:446): pid=7763 uid=0 auid=4294967295 ses=4294967295 msg='op=destroy
type=CRYPTO_KEY_USER msg=audit(1485857482.533:447): pid=7763 uid=0 auid=4294967295 ses=4294967295 msg='op=destroy
type=CRYPTO_KEY_USER msg=audit(1485857482.533:448): pid=7763 uid=0 auid=4294967295 ses=4294967295 msg='op=destroy
type=CRYPTO_SESSION msg=audit(1485857482.533:449): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=start dir
? addr=192.168.66.1 terminal=? res=success'
type=CRYPTO_SESSION msg=audit(1485857482.533:450): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=start dir
? addr=192.168.66.1 terminal=? res=success'
type=USER_AUTH msg=audit(1485857482.597:451): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=pubkey_auth rp
type=USER_AUTH msg=audit(1485857482.597:452): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=key algo=ssh-r
type=USER_ACCT msg=audit(1485857482.597:453): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=PAM:accounting
type=CRYPTO_KEY_USER msg=audit(1485857482.598:454): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=destroy
type=USER_AUTH msg=audit(1485857482.599:455): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=success acct="
type=CRED_ACQ msg=audit(1485857482.600:456): pid=7762 uid=0 auid=4294967295 ses=4294967295 msg='op=PAM:setcred gra
type=LOGIN msg=audit(1485857482.600:457): pid=7762 uid=0 old-auid=4294967295 auid=54321 old-ses=4294967295 ses=3 r
type=USER_START msg=audit(1485857482.606:458): pid=7762 uid=0 auid=54321 ses=3 msg='op=PAM:session_open grantors=p
erminal=ssh res=success'
type=CRYPTO_KEY_USER msg=audit(1485857482.606:459): pid=7762 uid=0 auid=54321 ses=3 msg='op=destroy kind=session f
type=CRYPTO_KEY_USER msg=audit(1485857482.607:460): pid=7764 uid=0 auid=54321 ses=3 msg='op=destroy kind=server fp
type=CRYPTO_KEY_USER msg=audit(1485857482.607:461): pid=7764 uid=0 auid=54321 ses=3 msg='op=destroy kind=server fp
type=CRYPTO_KEY_USER msg=audit(1485857482.607:462): pid=7764 uid=0 auid=54321 ses=3 msg='op=destroy kind=server fp
type=CRED_ACQ msg=audit(1485857482.607:463): pid=7764 uid=0 auid=54321 ses=3 msg='op=PAM:setcred grantors=pam_env,
type=USER_LOGIN msg=audit(1485857482.608:464): pid=7762 uid=0 auid=54321 ses=3 msg='op=login id=54321 exe="/usr/sb
type=USER_START msg=audit(1485857482.608:465): pid=7762 uid=0 auid=54321 ses=3 msg='op=login id=54321 exe="/usr/sb
type=CRYPTO_KEY_USER msg=audit(1485857482.609:466): pid=7762 uid=0 auid=54321 ses=3 msg='op=destroy kind=server fp
```


LINUX KERNEL AUDIT FRAMEWORK

- This by default does NOT solve the database stop/start audit issue.

- With a simple addition of audit rules, it's possible to gain insight into clusterware actions:

```
# auditctl -w /u01/app/12.1.0.2/grid/bin/srvctl -p x -k oracrs
# auditctl -w /u01/app/12.1.0.2/grid/bin/crsctl -p x -k oracrs
```

- This puts a 'watch' (-w) on the execution (-p x) of the the srvctl and crsctl executables.
  - Non intrusive, there is no change in behaviour of using the executables.
  - However, upon execution, the audit daemon logs who executed the watched executable.
  - The audit results are marked with the 'oracrs' key.

- Above adds rules on runtime, in order to make the audit rules addition permanent, add the rules to /etc/audit/audit.rules, starting from '-w'.

LINUX KERNEL AUDIT FRAMEWORK

- Once audit rules are set, this is how to use them for inspecting the audit results:

```
# ausearch -i -k oracrs
----
type=CONFIG_CHANGE msg=audit(01/31/2017 10:40:43.860:235104) : auid=opc ses=7134
op="add_rule" key=oracrs list=exit res=yes
----
type=CONFIG_CHANGE msg=audit(01/31/2017 10:40:54.583:235105) : auid=opc ses=7134
op="add_rule" key=oracrs list=exit res=yes
```

LINUX KERNEL AUDIT FRAMEWORK

```
type=PROCTITLE msg=audit(01/31/2017 10:45:28.310:235141) : proctitle=/bin/sh /
u01/app/12.1.0.2/grid/bin/srvctl start database -d aobg
type=PATH msg=audit(01/31/2017 10:45:28.310:235141) : item=2 name=/lib64/ld-
linux-x86-64.so.2 inode=2097766 dev=08:03 mode=file,755 ouid=root ogid=root
rdev=00:00 nametype=NORMAL
type=PATH msg=audit(01/31/2017 10:45:28.310:235141) : item=1 name=/bin/sh
inode=524291 dev=08:03 mode=file,755 ouid=root ogid=root rdev=00:00
nametype=NORMAL
type=PATH msg=audit(01/31/2017 10:45:28.310:235141) : item=0 name=/u01/app/
12.1.0.2/grid/bin/srvctl inode=201328006 dev=fb:00 mode=file,750 ouid=oracle
ogid=oinstall rdev=00:00 nametype=NORMAL
type=CWD msg=audit(01/31/2017 10:45:28.310:235141) :  cwd=/home/oracle
type=EXECVE msg=audit(01/31/2017 10:45:28.310:235141) : argc=6 a0=/bin/sh a1=/
u01/app/12.1.0.2/grid/bin/srvctl a2=start a3=database a4=-d a5=aobg
type=SYSCALL msg=audit(01/31/2017 10:45:28.310:235141) : arch=x86_64
syscall=execve success=yes exit=0 a0=0x201c270 a1=0x200b5b0 a2=0x20296e0
a3=0x7ffc591fef80 items=3 ppid=3721 pid=3751 auid=opc uid=oracle gid=oinstall
euid=oracle suid=oracle fsuid=oracle egid=oinstall sgid=oinstall fsgid=oinstall
tty=pts1 ses=7164 comm=srvctl exe=/bin/bash key=oracrs
```


LINUX KERNEL AUDIT FRAMEWORK

```
# ausearch --session 7164 -i | grep -e CRED_ACQ -e USER_LOGOUT -e EXECVE
type=CRED_ACQ msg=audit(01/31/2017 10:44:46.998:235125) : pid=3685 uid=root
auid=opc ses=7164 msg='op=PAM:setcred acct=opc exe=/usr/sbin/sshd
hostname=316.93.231.22 addr=316.93.231.22 terminal=ssh res=success'
type=CRED_ACQ msg=audit(01/31/2017 10:45:15.589:235132) : pid=3719 uid=root
auid=opc ses=7164 msg='op=PAM:setcred acct=root exe=/usr/bin/sudo hostname=?
addr=? terminal=/dev/pts/1 res=success'
type=CRED_ACQ msg=audit(01/31/2017 10:45:15.593:235137) : pid=3720 uid=root
auid=opc ses=7164 msg='op=PAM:setcred acct=oracle exe=/bin/su hostname=? addr=?
terminal=pts/1 res=success'
type=EXECVE msg=audit(01/31/2017 10:45:28.310:235141) : argc=6 a0=/bin/sh a1=/
u01/app/12.1.0.2/grid/bin/srvctl a2=start a3=database a4=-d a5=aobg
type=USER_LOGOUT msg=audit(01/31/2017 10:45:44.899:235147) : pid=3682 uid=root
auid=opc ses=7164 msg='op=login id=opc exe=/usr/sbin/sshd hostname=? addr=?
terminal=/dev/pts/1 res=success'
```

LINUX KERNEL AUDIT - EXADATA

- In fact, Oracle applies audit rules on Exadata compute nodes! These are the audit keys that are set:
  - time-change: executables and file that affect time.
  - identity: files and executables that influence identity.
  - system-locale: /etc/ issue, issue.net, hosts and /etc/sysconfig/network.
  - MAC-policy: changes to /etc/selinux/.
  - logins: changes to /var/log/lastlog and /var/log/faillog.
  - session: changes to /var/log/utmp bump wtmp.
  - perm_mod: any system call that changes permission.
  - access: any unauthorised file access (-EACCESS and -EPERM).
  - export: execution of the mount command to mount media.
  - delete: any successful or unsuccessful deletion.
  - actions: changes to /etc/sudoers.
  - modules: kernel module manipulation.
  - power: shutdown, power off, reboot and halt execution.
  - privileged: setuid and setgid executables, both O/S and oracle.

LINUX KERNEL AUDIT - EXADATA

- However, only SUID/SGID Oracle executables are audited.
  - srvctl and crsctl are not audited, which are used to stop/start any clusterware managed component or even clusterware itself (!).
  - I strongly advise to add a watch for execution these two files so a user-initiated change to clusterware and its managed resources is audited and logged.

LINUX KERNEL AUDIT - EXADATA

- **Exadata contains a set of audit rules which are added by Oracle.**
  The audit rules are not exhaustive, not ‘everything’ is visible in the audit log.

- **Here’s an example of the usefulness of audit rules:**

  Case: server got rebooted. Obviously, reason unknown.

LINUX KERNEL AUDIT - EXADATA

- First point to look at, is the /var/log/messages file.

  Often, the only indication of a reboot is the kernel indicating it is starting:

```
# grep '/proc/kmsg started' messages
Feb  6 20:43:19 enkdb03 kernel: imklog 5.8.10, log source = /proc/kmsg started.
```

- Okay, so startup time is Feb 6, 20:43. Let's see what the last logged lines are before the reboot/kernel start:

```
# grep -B 50 '/proc/kmsg started' messages
…
Feb  6 20:39:53 enkdb03 kernel: RDS: rds_bind() could not find a transport, load rds_t
Feb  6 20:39:57 enkdb03 kernel: rds_bind: 27 callbacks suppressed
Feb  6 20:40:00 enkdb03 kernel: RDS: rds_bind() could not find a transport, load rds_t
Feb  6 20:43:19 enkdb03 kernel: imklog 5.8.10, log source = /proc/kmsg started.
```


LINUX KERNEL AUDIT - EXADATA

- Often, a reboot of an Oracle cluster node happens because of cluster consistency reasons.

  The way for a cluster to remain healthy is by removing unhealthy nodes (STONITH).

  The cluster daemons also can decide to commit suicide if it understands it lost contact with the entire cluster.

```
# grep -B1 CSS /u01/app/11.2.0.3/grid/log/enkdb03/alertenkdb03.log
2017-02-06 20:40:00.561:
[cssd(184549)]CRS-1609:This node is unable to communicate with other nodes in the cluster and
is going down to preserve cluster integrity; details at (:CSSNM00008:) in /u01/app/11.2.0.3/
grid/log/enkdb03/cssd/ocssd.log.
2017-02-06 20:40:00.561:
[cssd(184549)]CRS-1656:The CSS daemon is terminating due to a fatal error; Details at
(:CSSSC00012:) in /u01/app/11.2.0.3/grid/log/enkdb03/cssd/ocssd.log
--
[ohasd(26964)]CRS-8011:reboot advisory message from host: enkdb03, component: cssmonit, with
time stamp: L-2017-02-06-20:40:07.714
[ohasd(26964)]CRS-8013:reboot advisory message text: clsnomon_status: need to reboot,
unexpected failure 8 received from CSS
```

LINUX KERNEL AUDIT - EXADATA

- So the conclusion at this point is simple:
  (infiniband) network gone, CSS rebooted the node in order to (try to) restore cluster communication.

- The obvious question now is: why did the network go down?
  Look in the linux messages file again (backwards starting from kernel start line):

```
# less /var/log/messages
Feb  6 20:39:11 enkdb03 kernel: RDS: rds_bind() could not find a transport, load rds_tcp
or rds_rdma?
Feb  6 20:39:00 enkdb03 kernel: bonding: bondib0: releasing active interface ib1
Feb  6 20:39:00 enkdb03 kernel: bonding: bondib0: Removing slave ib1.
Feb  6 20:39:00 enkdb03 kernel: bonding: bondib0: making interface ib1 the new active one.
Feb  6 20:39:00 enkdb03 kernel: bonding: bondib0: releasing active interface ib0
Feb  6 20:39:00 enkdb03 kernel: bonding: bondib0: Removing slave ib0.
```

LINUX KERNEL AUDIT - EXADATA

- **Linux released the interfaces from the bond ITSELF!**
  That's weird?!

- **Always carefully inspect everything surrounding your problem.**
  Don't stop at the problem itself.

LINUX KERNEL AUDIT - EXADATA

- A little further UP the log file (=back in time) I found this:

```
# less /var/log/messages
Feb  6 20:39:00 enkdb03 kernel: bonding: bondeth0: releasing active interface eth5
Feb  6 20:39:00 enkdb03 kernel: bonding: bondeth0: Removing slave eth5.
Feb  6 20:38:59 enkdb03 kernel: bonding: bondeth0: releasing active interface eth4
Feb  6 20:38:59 enkdb03 kernel: bonding: bondeth0: Warning: the permanent HWaddr of
eth4 - 90:e2:ba:3e:22:f4 - is still in use by bondeth0. Set the HWaddr of eth4 to a
different address to avoid conflicts.
Feb  6 20:38:59 enkdb03 kernel: bonding: bondeth0: Removing slave eth4.
```

- This is the kernel releasing the interfaces from the ethernet (client network) bond too!

  This is not an infiniband problem, this looks like the network was stopped!

LINUX KERNEL AUDIT - EXADATA

- So, is there something we can see using auditing?
  - Yes and no.
  - On a default linux system, you can't.
  - On Exadata, there are the Oracle provided audit rules.
  - However, these do not include stopping or starting of linux daemons.

- I first investigated the audit records exactly on the time the network change occurred.

```
# ausearch -i -ts 02/06/2017 20:38:00 -te 02/06/2017 20:42:00
```

- (that's dd/MM/yyyy hh:mm:ss)

LINUX KERNEL AUDIT - EXADATA

- **Audit results that are relevant to the issue:**

```
…lots of output…
type=PATH msg=audit(02/06/2017 20:39:01.853:10660) : item=1 name=/var/lock/subsys/
network inode=3555635 dev=fc:00 mode=file,644 ouid=root ogid=root rdev=00:00
type=PATH msg=audit(02/06/2017 20:39:01.853:10660) : item=0 name=/var/lock/subsys/
inode=3555434 dev=fc:00 mode=dir,755 ouid=root ogid=root rdev=00:00
type=CWD msg=audit(02/06/2017 20:39:01.853:10660) :  cwd=/etc/sysconfig/network-
scripts
type=SYSCALL msg=audit(02/06/2017 20:39:01.853:10660) : arch=x86_64 syscall=unlinkat
success=yes exit=0 a0=0xffffffffffffff9c a1=0xa420c0 a2=0x0 a3=0x7ffdf67ae5a0 items=2
ppid=82319 pid=86567 auid=xxxxxx uid=root gid=root euid=root suid=root fsuid=root
egid=root sgid=root fsgid=root tty=pts1 ses=626 comm=rm exe=/bin/rm key=delete
…lots of output…
```


LINUX KERNEL AUDIT - EXADATA

- On OL6, the (old) SystemV run level scripts are in /etc/rc.d/init.d.
  - SystemV run level scripts touch a file in /var/lock/subsys with the name of the service they start, and remove that file on shutdown of the service.
  - The service stop/start/restart is not visible in the audit data, but the removal of the file in /var/lock/subsys is.
  - Inside the network script, the CWD is explicitly set, which is visible in the audit data too.
  - The date and time matches too: the network service script removes the /var/lock/subsys/network file AFTER it stopped it all.
  - So now we can look at the session number, and see log-in time, ip address and the sudo action to get a full picture of what happened in the session.

LINUX KERNEL AUDIT - EXADATA

- The session number was 626.

```
# ausearch --session 626 -i | grep -e CRED_ACQ -e USER_LOGOUT -e EXECVE -e key=delete
type=CRED_ACQ msg=audit(02/06/2017 20:38:20.992:10649) : user pid=25300 uid=root
auid=xxxxxx ses=626 msg='op=PAM:setcred acct=xxxxxx exe=/usr/sbin/sshd
hostname=amsmon00.ux.cba.org addr=10.10.10.10 terminal=ssh res=success'
type=CRED_ACQ msg=audit(02/06/2017 20:38:24.464:10658) : user pid=30043 uid=root
auid=xxxxxx ses=626 msg='op=PAM:setcred acct=root exe=/usr/bin/sudo hostname=? addr=?
terminal=/dev/pts/1 res=success'
type=SYSCALL msg=audit(02/06/2017 20:39:01.853:10660) : arch=x86_64 syscall=unlinkat
success=yes exit=0 a0=0xffffffffffffff9c a1=0xa420c0 a2=0x0 a3=0x7ffdf67ae5a0 items=2
ppid=82319 pid=86567 auid=xxxxxx uid=root gid=root euid=root suid=root fsuid=root
egid=root sgid=root fsgid=root tty=pts1 ses=626 comm=rm exe=/bin/rm key=delete
```

PANNING OUT - THE BIG PICTURE

- The Oracle (database and ASM instances) log into .aud files.
  - These files are owned by the same user as probably is used to manipulate a database or ASM instance.
  - This means that if someone wants to hide any actions, this is simply doable by removing selected or all audit files.
  - In fact, you should automatically remove audit files because the number grows high real fast!
- If you need reliable Oracle logging, there are a couple of things you can do.
  1. Set AUDIT_SYSLOG_LEVEL to make an instance write the audit records to syslog.
     - This prevents the creation of .aud files.
     - This prevents the oracle owner from being able to manipulate logging data.
     - *Do not forget to configure a separate log in rsyslog.conf to prevent SYSDBA logging from massively polluting the messages file.*
  2. Have a log data shipper (like filebeat, logstash, etc.) immediately send new information outside of the current machine.

PANNING OUT - THE BIG PICTURE

- On the operating system level, essentially the same issue exists:
  - The root user can manipulate and or remove any file.
  - Because there are multiple files (/var/log/secure, /var/log/wtmp, /var/log/audit/audit.log, /var/log/lastlog), and some are not text format, this might be harder. But it's absolutely not impossible.

- Solutions:
  1. Have a log data shipper (like filebeat, logstash, etc.) immediately send new information outside of the current machine. (to which the machine admin has no root access).
  2. Remote syslogging: configure syslog to send logs to a remote server.
     - Please mind you need to make sure ALL relevant data goes through syslog.
     - Linux auditing can be configured to send output to syslog.

SUMMARY

- The Oracle database provides audit records to indicate who performed shutdown/startup.
  - These can be used to do an investigation into who/why a shutdown happened.
  - However, these can easily be manipulated.

- Linux provides sophisticated auditing with the kernel audit facility.
  - Still, the information is stored on the very same machine that is audited.
  - Harder to manipulate, but absolutely not impossible.
  - Need to manipulate multiple files to leave no trace.
    - Probably the only rule that can detect local tampering is a watch on /var/log/audit/audit.log.

- One of the goals for this presentation is to provide information on how you can trace a user session using audit information available by default.

SUMMARY

- If you are serious about auditing, and need to keep that data, you need to move audit data off the local machine to prevent tampering.

- Oracle clusterware usage means you can't audit user-initiated changes to the cluster and cluster resources.
  - Unless you put an 'execution watch' on crsctl and srvctl.

