Webinar
Oracle Database 11g: Security

Uwe Hesse

Oracle University Principal Instructor

More about me: http://uhesse.wordpress.com/about

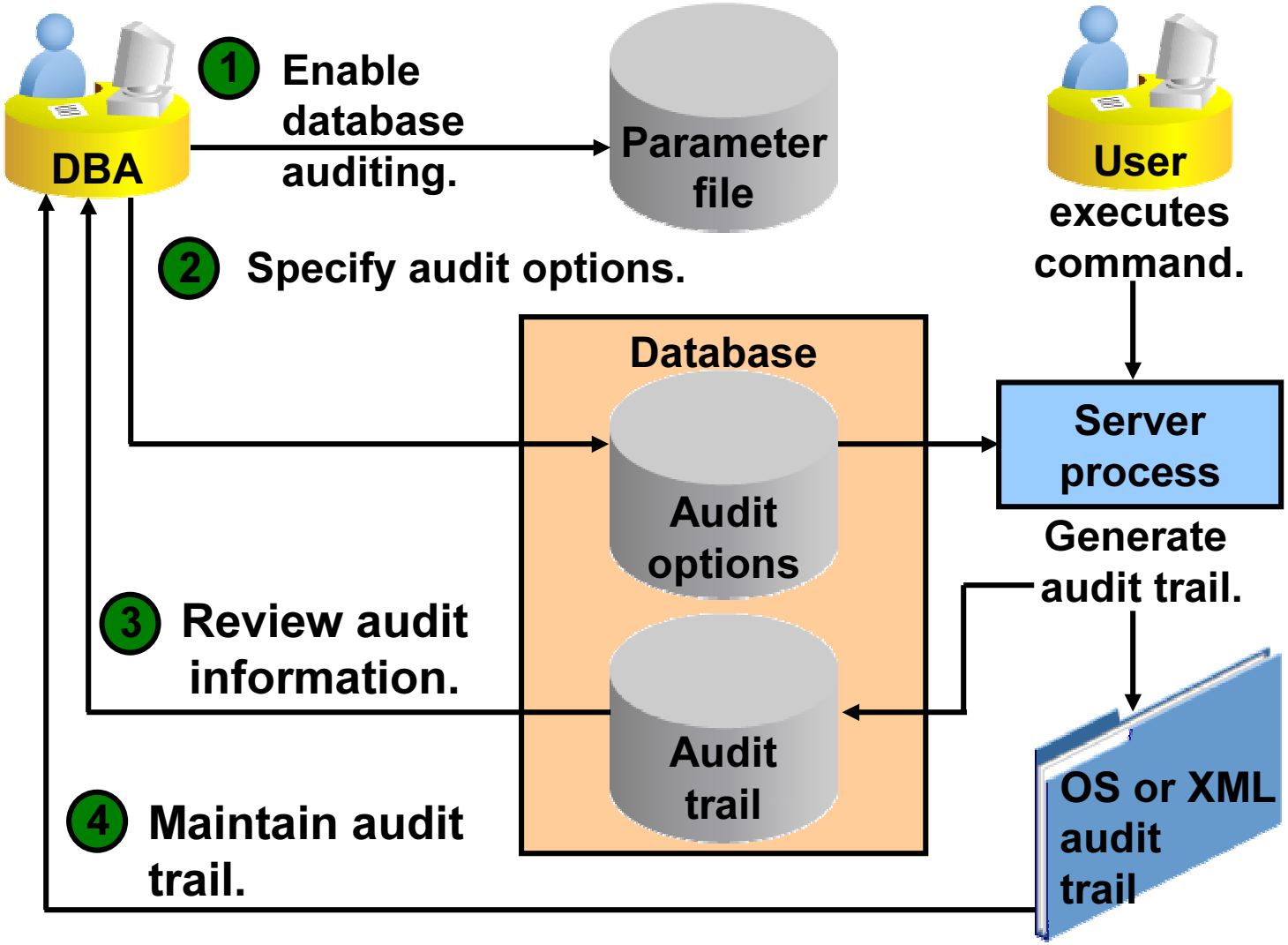
Get these slides: http://uhesse.wordpress/downloads

ORACLE

Agenda

- "Traditional" Auditing
  - Brief introduction
  - Demonstration
- Auditing the `SYS` user
  - Brief introduction
  - Demonstration
- Fine Grained Auditing
  - Brief introduction
  - Demonstration
- Transparent Data Encryption
  - Brief introduction
  - Demonstration

“Traditional” Auditing



Examples for "Traditional" auditing on: a) Privileges, b) Objects, c) Statements

```
SQL> alter system set audit_trail='DB','EXTENDED'
   scope=spfile;
```

```
SQL> audit session whenever not successful; -- a)
```

```
SQL> audit all on hr.employees by access whenever
   successful; -- b)
```

```
SQL> audit table by access whenever successful; -- c)
```

Demonstration
Traditional Auditing ...

Possible values for AUDIT_TRAIL:

NONE (Default before 11g)

DB (Default in 11g) → Table sys.aud$, accessed by DBA_AUDIT_TRAIL

OS → OS textfile, directory specified by AUDIT_FILE_DEST

XML → OS file, dito, accessed by V$XML_AUDIT_TRAIL

DB, EXTENDED (setting in our demo, includes *complete statements*)

XML, EXTENDED (as above, in XML files)

Auditing the SYS user

Users with the SYSDBA or SYSOPER privileges can connect when the database is closed.

- **Audit trail *must* be stored outside the database**
- **Directory is determined by audit_file_dest**
- ***Connections* as SYSDBA or SYSOPER are *always audited.***
- **You can enable *additional auditing* with audit_sys_operations**

Steps to audit all actions of `sys` into a file owned by `root`

```
SQL> alter system set audit_sys_operations = true
    scope=spfile;
```

```
SQL> alter system set audit_syslog_level =
    'LOCAL1.WARNING' scope=spfile;
```

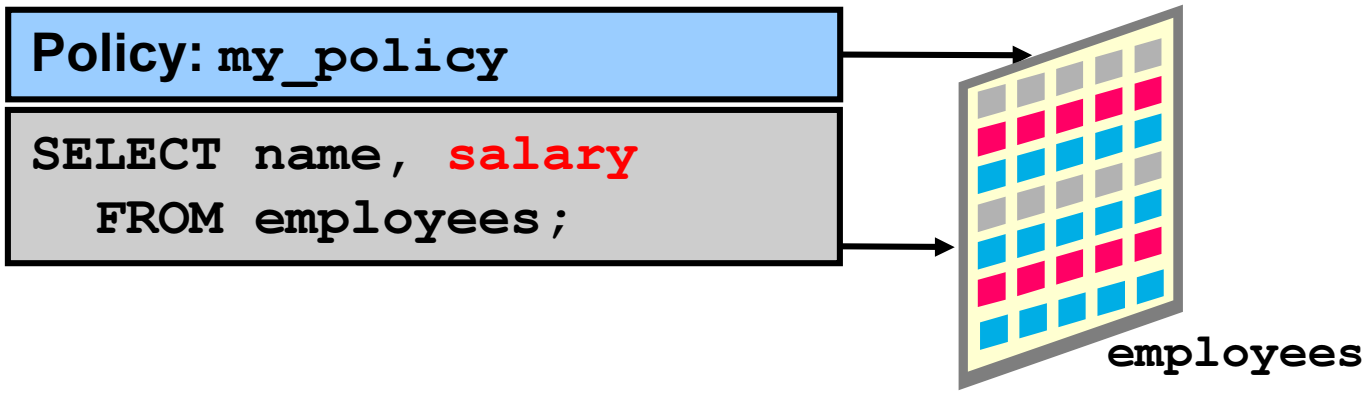
```
[root@uhesse ~]# cat /etc/syslog.conf | grep local1
local1.warning /var/log/audit.log
```

```
[root@uhesse ~]# /etc/rc.d/init.d/syslog restart
```

Demonstration
Auditing sys ...

Fine-Grained Auditing

- Audits *only if definable conditions are met*
- Possible for `SELECT` and DML
- Can be linked to a table or view, to one or more columns
- Implemented with the `DBMS_FGA` package



FGA Policy in action:

```
BEGIN
  dbms_fga.add_policy(
  object_schema =>    'HR',
  object_name =>      'EMPLOYEES',
  policy_name =>      'my_policy',
  audit_column =>     'salary',
  statement_types => 'SELECT,INSERT,UPDATE,DELETE');
END;
/
```

```
SQL> select last_name,email from employees; -- no audit entries!
```

```
SQL> select last_name,salary from employees; -- audited!
SQL> update employees set salary = salary+500; -- audited!
```

```
SQL> select db_user,os_user,sql_text,timestamp
     from dba_fga_audit_trail where db_user='HR';
```

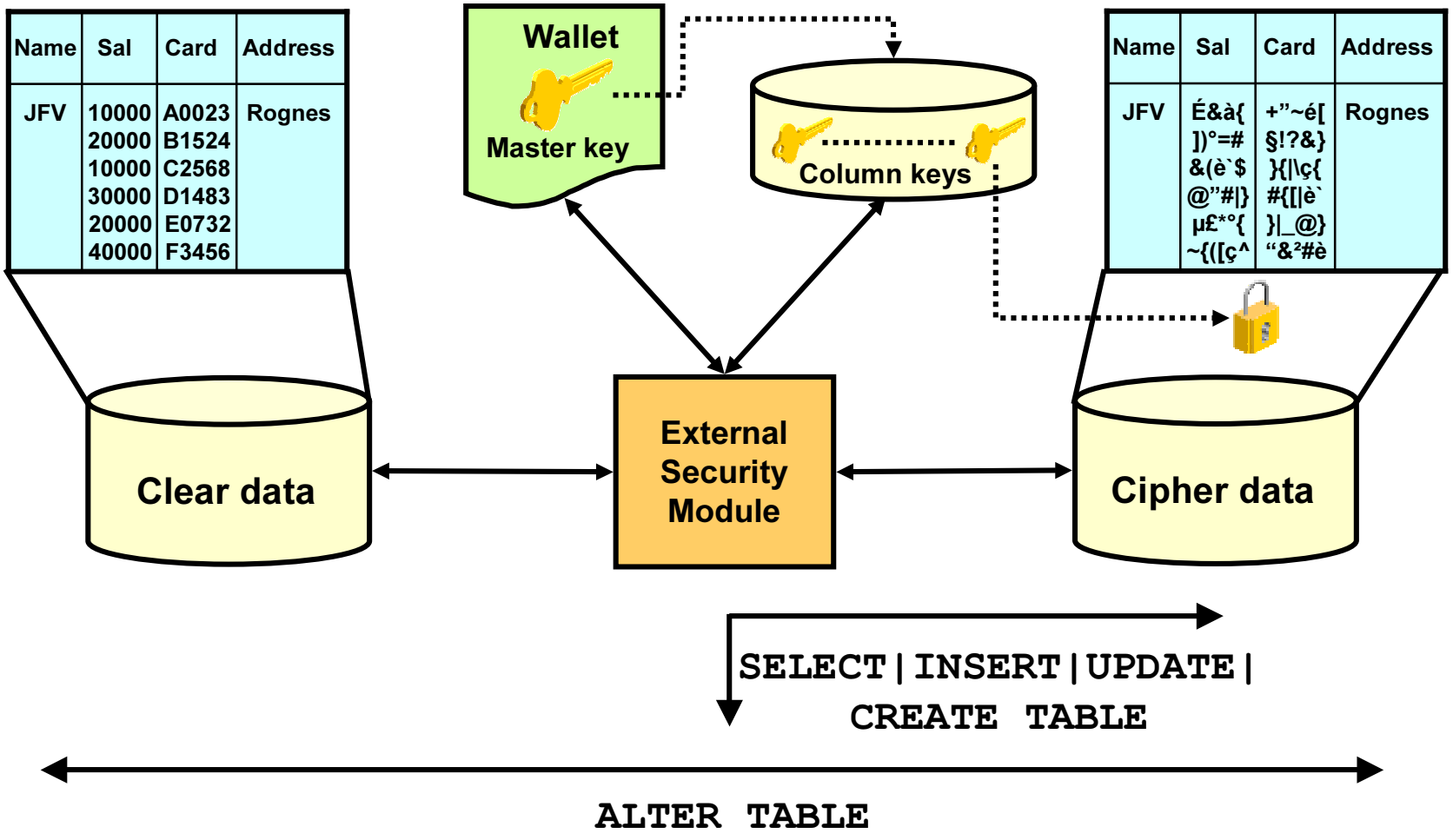
Demonstration
Fine Grained Auditing ...

Auditing Considerations

Some things to be aware of:

- **Are you in line with your (national) legal conditions?**
- **Auditing may consume much space, especially**
  - **with `AUDIT_TRAIL = ..., EXTENDED`**
  - **with `AUDIT_SYS_OPERATIONS = TRUE`**
- **A strategy is needed to**
  - ***analyse* the audit information**
  - ***remove* old audit information**
    - **from `sys.aud$` (Traditional Auditing)**
    - **from filesystem (auditing of `sys` or `AUDIT_TRAIL=OS`)**
    - **from `sys.fga_log$` (Fine Grained Auditing)**

Transparent Data Encryption: Overview



Steps to setup TDE ...

Add the following to `$ORACLE_HOME/network/admin/sqlnet.ora` :

```
ENCRYPTION_WALLET_LOCATION=
(source=
      (METHOD=FILE)
        (METHOD_DATA=
        (DIRECTORY=/u01/app/oracle/admin/orcl/wallet)
      )
  )
```

Create that directory at the OS prompt:

```
mkdir /u01/app/oracle/admin/orcl/wallet
```

Set the TDE master password (implictly creating the wallet):

```
SQL> alter system set encryption key identified by "oracle";
```

... continued Steps TDE

In 10g already: Create tables with encrypted columns

```
SQL> create table cia.bgents
   (id number, covername varchar2(20), realname
   varchar2(20) encrypt);
```

In 11g: Create encrypted tablespaces:

```
SQL> create tablespace krypto datafile
   '/u01/app/oracle/oradata/orcl/krypto01.dbf' size 1m
   encryption default storage (encrypt) ;
```

TDE in action...

Attacker has stolen my whole database!

After startup of the instance:

```
SQL>  select * from cia.bgents;
 select * from cia.bgents
                    *
ERROR at line 1:
ORA-28365: wallet is not open
```

Let‘s look into the datafile:

```
[oracle@uhesse orcl]$ strings krypto01.dbf | grep Hesse
```

It‘s encrypted! No realname is exposed. (10g and 11g way either)

... continued TDE in action

What if we dump the blocks?

```
SQL> select dbms_rowid.rowid_relative_fno(rowid),
   dbms_rowid.rowid_block_number(rowid) from cia.bgents;

SQL> alter system dump datafile &x block &y;
```

The block dumps in `user_dump_dest` don‘t expose the encrypted values! Only if we specify the *correct password of the original wallet*, the encrypted content is readable:

```
SQL> alter system set encryption wallet open identified
   by "oracle";
```

Even after the creation of a new wallet, it would not be possible to read the encrypted tables!

Demonstration
Transparent Data Encryption ...

TDE Considerations

- **You need a backup of the wallet! It can't be simply recreated ...**
- **TDE does not protect against DB users, that have access privileges on the tables. That is exactly the meaning of "Transparent" here.**

Thank you for your attention!

Get these slides: http://uhesse.wordpress/downloads