ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

Самарский национальный исследовательский

университет имени академика С. П. Королева

Кафедра программных систем

Дисциплина: «Безопасность систем баз данных»

Отчёт

по лабораторной работе № 24

«Шифрование данных в СУБД *Oracle*»

Выполнила:

студентка группы 6411 100503D

Стерпу Е.К.

Проверил:

Додонов М.В.

Самара

2018

**Цель:** обучение методам и средствам шифрования данных, хранящихся в базах данных СУБД *Oracle.*

#### ***Последовательность действий обучаемого***

1. Настроить систему для применения *TDE*. Определить место хранения бумажника на устройстве *J:/*.
2. Создать бумажник с возможностью доступа к нему по паролю *walletpsw*. Убедиться, что бумажник создан (на устройстве появился новый файл):

sterpu> create temporary tablespacetemp01

2 tempfile 'c:\lr\_24\temp01.dbf' size 5m

3 extent management local uniform size 100k;

sterpu> create tablespace data01

2 datafile 'c:\lr\_24\data01.dbf' size 10m;

sterpu> create user sec identified by 000

2 default tablespace data01

3 temporary tablespace temp01

4 quota 5m on data01;

sterpu> grant create session, create table, create procedure, alter system to sec;

sterpu> alter system set encryption key identified by "walletpsw";

1. Создать и заполнить таблицу, изменить спецификации одного столбца на шифрованный:

sterpu> conn sec/000

sterpu> create table tab1(at1 number, at2 varchar2(30));

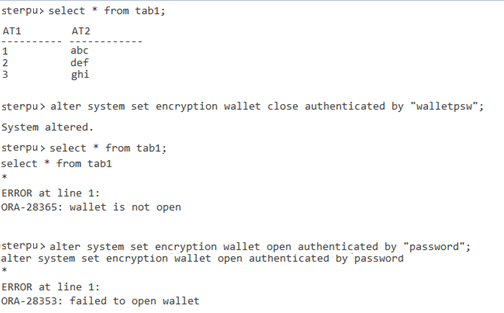
sterpu> insert into tab1 values(1,'abc');

sterpu> insert into tab1 values(2,'def');

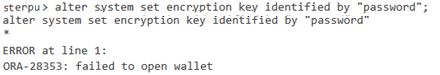
sterpu> insert into tab1 values(3,'ghi');

sterpu> alter table tab1 modify(at2 encrypt);

1. Открыть бумажник, проверить возможность чтения открытой информации. Закрыть бумажник, убедиться, что не доступна зашифрованная информация. Убедиться, что открыть бумажник с неверным паролем нельзя.



1. Попытаться создать «логическую копию» бумажника. Убедиться, что это не приводит к возможности получить доступ к данным:



1. Создать процедуру шифрования данных столбца таблицы с использованием средств пакета *DBMS*\_*CRYPTO*. В тексте этой процедуры ключ шифрования должен быть представлен константой, а секретность технологии шифрования - определяться знанием имени процедуры и получением привилегии на ее исполнение. В качестве криптографического алгоритма выбрать *AES* с длиной ключа 128 бит.

sterpu> conn sys/1 as sysdba

sterpu> grant execute on dbms\_crypto to sec;

sterpu> conn sec/000

sterpu> create table tab\_sec(at1 varchar2(30), at2 raw(2000));

sterpu> create or replace procedure encrypt\_tab(input\_string varchar2) as

2 encrypted\_raw raw(2000);

3 encryption\_key\_str varchar2(16):='sec\_key030030416';

4 encryption\_key raw(16);

5 encryption\_type pls\_integer:=dbms\_crypto.encrypt\_aes128

6 +dbms\_crypto.chain\_cbc+dbms\_crypto.pad\_pkcs5;

7 begin

8 encryption\_key:=utl\_raw.cast\_to\_raw(encryption\_key\_str);

9 encrypted\_raw:=dbms\_crypto.encrypt

10 (utl\_i18n.string\_to\_raw(input\_string,'al32utf8'),

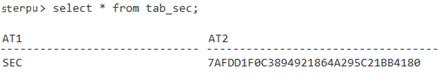
11 encryption\_type, encryption\_key);

12 insert into tab\_sec values(user, encrypted\_raw);

13 end;

14 /

sterpu> exec encrypt\_tab('proverka');



sterpu> delete from tab\_sec;

1. Реализовать процедуру шифрования с внешним хранением ключей. Ключи хранить в формате *RAW*. В качестве места хранения ключа выбрать устройство флэш-памяти. Для генерации значений ключа воспользоваться процедурой *RANDOMBYTES* из пакета *DBMS*\_*CRYPTO*. Для задания начального значения генератора можно использовать параметр *SQLNET*.*CRYPTO*\_*SEED* файла *SQLNET*.*ORA*, который может принимать значения длиной от 10 до 70 байт. Запись данных на устройство флэш-памяти осуществить процедурой Запись данных на устройство флэш-памяти осуществить процедурой *PUT*\_*RAW* пакета *UTL*\_*FILE*.

sterpu> conn sys/1 as sysdba

sterpu> create directory key\_dir as 'd:\';

sterpu> grant read, write on directory key\_dir to sec;

sterpu> conn sec/000

sterpu> declare

2 key\_raw raw(16);

3 fk1 utl\_file.file\_type;

4 begin

5 fk1:=utl\_file.fopen('KEY\_DIR','key1','wb',16);

6 key\_raw:=dbms\_crypto.randombytes(16);

7 utl\_file.put\_raw(fk1,key\_raw);

8 utl\_file.fclose(fk1);

9 end;

10 /

sterpu> create or replace procedure encrypt\_tab\_pr(input\_string varchar2) as

2 encrypted\_raw raw(2000);

3 key\_raw raw(16);

4 encryption\_type pls\_integer:=dbms\_crypto.encrypt\_aes128

5 +dbms\_crypto.chain\_cbc+dbms\_crypto.pad\_pkcs5;

6 fk1 utl\_file.file\_type;

7 begin

8 fk1:=utl\_file.fopen('KEY\_DIR','key1','rb',16);

9 utl\_file.get\_raw(fk1,key\_raw);

10 utl\_file.fclose(fk1);

11 encrypted\_raw:=dbms\_crypto.encrypt(

12 utl\_i18n.string\_to\_raw(input\_string,'al32utf8'),

13 encryption\_type,key\_raw);

14 insert into tab\_sec values(user,encrypted\_raw);

15 end;

16 /

sterpu> create or replace function decrypt\_tab\_pr(encrypted\_raw raw)

2 return varchar2 as

3 decrypted\_string varchar2(200);

4 decrypted\_raw raw(2000);

5 key\_raw raw(16);

6 encryption\_type pls\_integer:=dbms\_crypto.encrypt\_aes128

7 +dbms\_crypto.chain\_cbc+dbms\_crypto.pad\_pkcs5;

8 fk1 utl\_file.file\_type;

9 begin

10 fk1:=utl\_file.fopen('KEY\_DIR','key1','rb',16);

11 utl\_file.get\_raw(fk1,key\_raw);

12 utl\_file.fclose(fk1);

13 decrypted\_raw:=dbms\_crypto.decrypt(encrypted\_raw,encryption\_type,key\_raw);

14 decrypted\_string:=utl\_i18n.raw\_to\_char(decrypted\_raw,'al32utf8');

15 dbms\_output.put\_line(decrypted\_string);

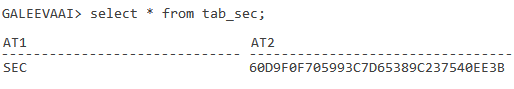
16 return decrypted\_string;

17 end;

18 /

sterpu> set serveroutput on

sterpu> exec encrypt\_tab\_pr('abcdefgh');



sterpu> declare test raw(200);

2 begin

3 select AT2 into test from tab\_sec;

4 dbms\_output.put\_line(decrypt\_tab\_pr(test));

5 end;

6 /