

西北工业大学

Northwestern Polytechnical University

数据库系统原理

Database System

第四章 数据库安全性

赵晓南

2024.10

- 实验课（头歌平台 & 本地环境）
- OceanBase技术报告与大赛报名
- 在线NPU-SQL-OJ平台



MiniOB · OceanBase 社区版

此次大赛分为初赛和决赛两个阶段。初赛基于 MiniOB 系统，门槛低，适合所有参赛选手。决赛基于 OceanBase，让选手体验真实的企业级数据库系统。MiniOB 是一个具备基础功能的项目实践型数据库，参赛选手可实现数据库的部分功能。初赛赛题由浅入深，帮助选手掌握基础入门数据库，逐步成为数据库内核“大神”。决赛是进阶实战挑战，选手在真实的企业级开源 OceanBase 内核上做更多数据库功能的探索。

2023 参赛时间 Important Dates

即日 — 23.10.25	23.10.17 — 23.11.06	23.11.09 — 23.12.11	2024年1月
报名	初赛	决赛	现场答辩
踊跃参与 所有高校学生均可报名	50 强 & 各省前 3 强 按总分评选全国 50 强及各省前 3 强	20 强 按总分评选全国 20 强	夺冠之夜 按得分及答辩评选全国 10 强

https://open.oceanbase.com/competition?_gl=1*v5nggf*_ga*MTc1NDIyMzE3OS4xNjk0NzY2MzAy*_ga_T35KTM57DZ*MTY5NTYxNDE2OC4yLjEUMTY5NTYxNDE2OC42MC4wLjA.#train



<https://www.oceanbase.com/video/9000654>

显示 5 项结果

搜索: 请输入查询关键字

测试ID	测试名称	发布时间	截止时间	开始人数	完成人数	平均完成率	平均正确率	详情
Exer-11	2023秋-练习1	2023年9月20日 12:50	2023年9月30日 23:00	29	17	54.84 %	95.92 %	

显示第 1 至 1 项结果, 共 1 项

数据查询语言

Select [ALL|DISTINCT] <输出属性列表>
From <一个或多个数据库表或视图>
[Where <查询条件>]
[Group By <分组条件>[HAVING <条件表达式>]]
[Order By <结果排序> [ASC|DESC]

1. 单表查询
2. 连接查询
3. 嵌套查询
4. 集合查询
5. 基于派生表的查询

SQL语言所使用的动词

SQL功能	动词
数据定义DDL	CREATE, DROP, ALTER
数据查询DQL	SELECT
数据更新DML	INSERT, UPDATE, DELETE
数据控制DCL	GRANT, REVOKE

4.1.计算机安全性概述

4.2.数据库安全性控制

4.3.视图机制

4.4.审计(Audit)

4.5.数据加密

4.6.其他安全性保护



◆ 数据库的安全性

指保护数据库，防止因用户非法使用数据库造成数据泄露、更改或破坏。

◆ 计算机系统安全性

- 技术安全
- 管理安全
- 政策法律

数据库安全与计算机系统的安全性，包括操作系统，网络系统的安全性紧密联系，相互支持



◆ 数据的完整性和安全性是两个不同概念

• 数据的完整性

➤ 防止数据库中存在不符合语义的数据，也就是防止数据库中存在不正确的数据

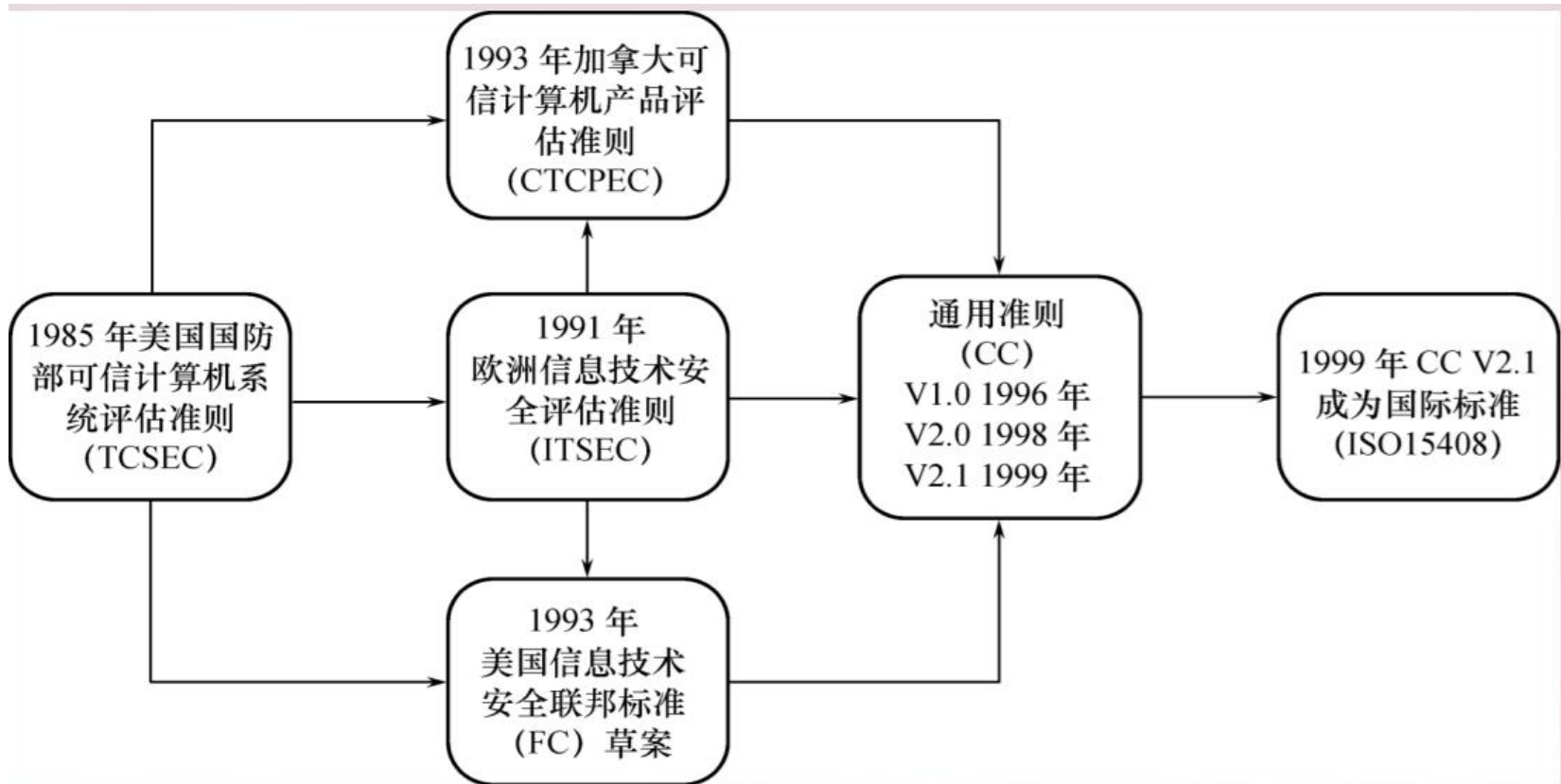
➤ 防范对象：不合语义的、不正确的数据

• 数据的安全性

➤ 保护数据库防止恶意的破坏和非法的存取

➤ 防范对象：非法用户和非法操作

4.1 计算机安全性概述



信息安全发展历程

《中华人民共和国数据安全法》是为了规范数据处理活动，保障数据安全，促进数据开发利用，保护个人、组织的合法权益，维护国家主权、安全和发展利益，制定的法律。

《中华人民共和国数据安全法》自()起施行。

- 1) 2020/9/1
- 2) 2021/9/1
- 3) 2022/9/1

本法所称**数据**，是指任何以电子或者其他方式对信息的记录。**数据处理**，包括数据的收集、存储、使用、加工、传输、提供、公开等。**数据安全**，是指通过采取必要措施，确保数据处于有效保护和合法利用的状态，以及具备保障持续安全状态的能力。

安全标准简介

- TCSEC (Trusted Computer System Evaluation Criteria, 桔皮书)

TDI (Trusted Database Interpretation, 紫皮书)

安全性级别划分标准：安全策略，责任，保证和文档

TDI
安全
级别
划分

安全级别	定义
A1	验证设计 (Verified Design)
B3	安全域 (Security Domains)
B2	结构化保护 (Structural Protection)
B1	标记安全保护 (Labeled Security Protection)
C2	受控的存取保护 (Controlled Access Protection)
C1	自主安全保护 (Discretionary Security Protection)
D	最小保护 (Minimal Protection)

TCSEC/TDI安全级别划分

安全指标	安全策略							责任			保证								文档				
	自主存取控制	客体重用	标记完整性	标记信息的扩散	主体敏感度标记	设备标记	强制存取控制	标识与鉴别	可信路径	审计	系统体系结构	系统完整性	屏蔽信道分析	可信设施管理	可信恢复	可安全测试	设计规范和验证	配置管理	可信分配	安全特性用户指南	可信设施手册	测试文档	设计文档
C1	■							■			■	■				■				■	■	■	■
C2	▨	■						▨		■	■	■				▨				■	▨	■	■
B1	■	■	■	■			■	▨		▨	▨	■				▨	■			■	▨	■	▨
B2	■	■		■	■	■	▨	■	■	▨	▨	■	■	■		▨	▨	■		■	▨	▨	▨
B3	▨	■				■	■	■	▨	▨	▨	■	▨	▨	■	▨	▨	■		■	▨	■	▨
A1		■		■		■	■	■	■	■		■	▨	■	■	▨	▨	▨	■	■	■	▨	▨

表 4.2 不同安全级别对安全指标的支持情况

 不支持

 支持较下一级有增/改

 同相邻低第一级

 新增支持

安全标准简介

- CC (Common Criteria) : 安全要求: 安全功能要求, 安全保证要求

CC
评估
保证
级
划
分

评估保 证级	定 义	TDI安全 级别
EAL1	功能测试 (functionally tested)	
EAL2	结构测试 (structurally tested)	C1
EAL3	系统地测试和检查 (methodically tested and checked)	C2
EAL4	系统地设计、测试和复查 (methodically designed, tested, and reviewed)	B1
EAL5	半形式化设计和测试 (semiformally designed and tested)	B2
EAL6	半形式化验证的设计和测试 (semiformally verified design and tested)	B3
EAL7	形式化验证的设计和测试 (formally verified design and tested)	A1

4.1.计算机安全性概述

4.2.数据库安全性控制

4.3.视图机制

4.4.审计(Audit)

4.5.数据加密

4.6.其他安全性保护



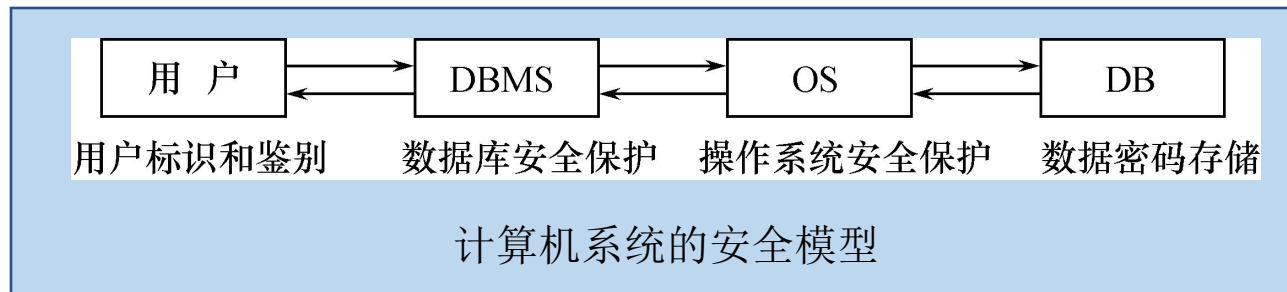
非法使用数据库的情况

- 编写一段合法的程序绕过DBMS及其授权机制；
- 直接或编写应用程序执行非授权操作；
- 通过多次合法查询数据库从中推导出一些保密数据

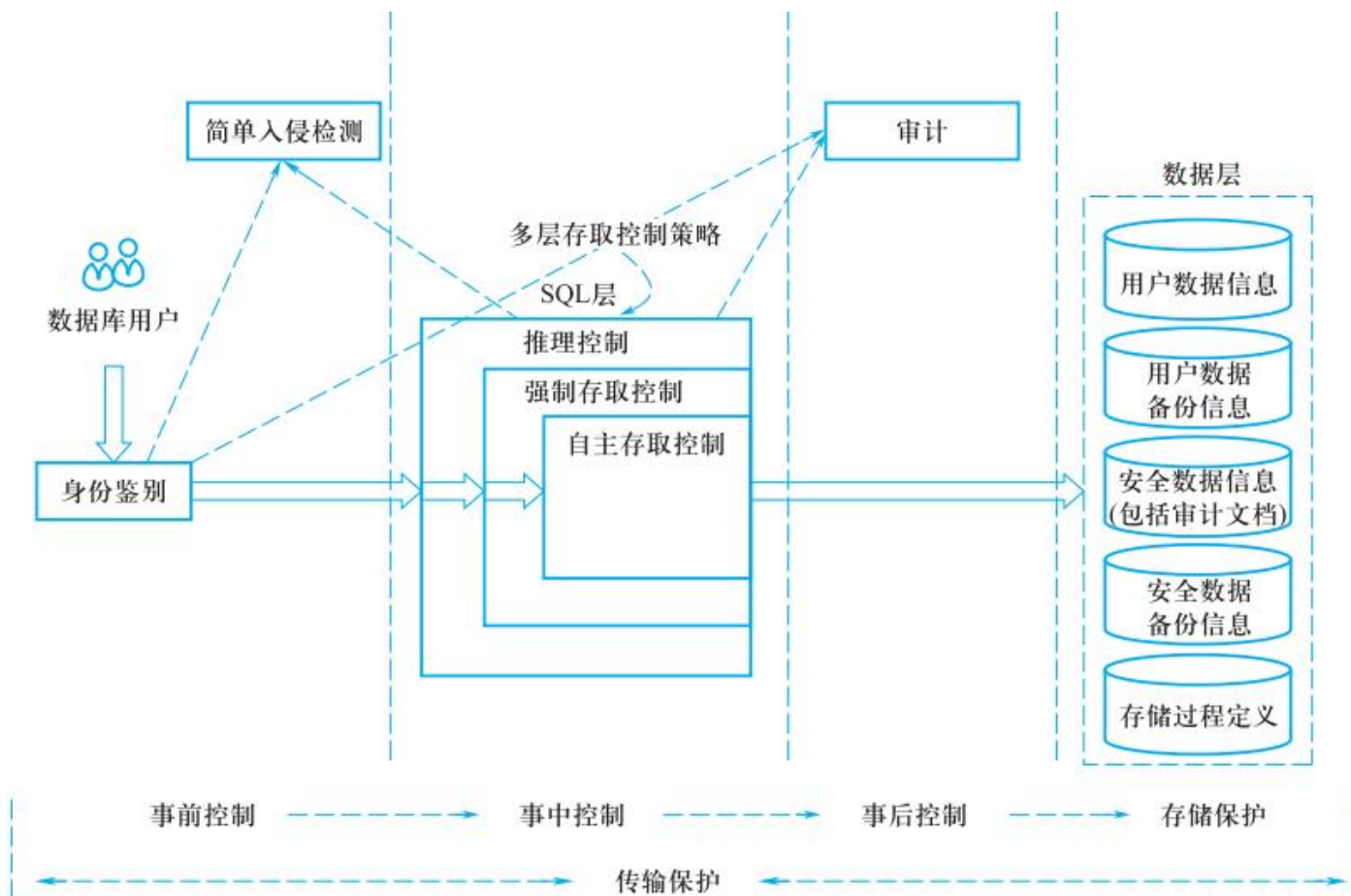
破坏安全性的行为可能是无意的，故意的，恶意的。

数据库安全性控制常用方法

- 用户标识和鉴定
- 存取控制
- 视图
- 审计
- 数据加密



4.2 数据库安全性控制



数据库管理系统的安全性控制模型

Application Security公司提出的十大数据库漏洞:

1. 默认、空白及弱用户名/密码
2. SQL注入
3. 广泛的用户和组权限
4. 启用不必要的数据库功能
5. 失效的配置管理
6. 缓冲区溢出
7. 特权升级
8. 拒绝服务攻击
9. 数据库未打补丁
10. 敏感数据未加密



<http://www.ok165.com/article/6670.html>

4.2 数据库安全性控制 — SQL注入



西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

- SQL Injection 的起因通常是因为程序采用了构造**字符串的方式**拼接SQL命令。
- 利用正常查询网站时，将**攻击指令藏于查询指令**中。
- 攻击者可穿透防火墙，绕过身份认证机制，取得资料库使用权限，进而窃取资料或窜改。破坏资料。



■ 一般输入帐号密码的网站SQL语法:

```
select * from member where UID = ' "& request("ID") &" ' ,  
and Passwd = ' "& request("Pwd") &" ' ,
```

✓ 如果正常使用者帐号A123456789 ， 密码1234

```
select * from member where UID = ' A123456789'  
and Passwd=' 1234'
```

通常输入的帐号和密码等信息会取代Web程序中的变量，并由两个单引号(' ')所包住。

若攻击者已知系统中已有一个Admin的管理者帐号，则输入Admin '—'，即可不须输入密码而进入资料库

```
select * from member where UID =' Admin '—',  
And Passwd =' '
```

注：— 符号后的任何叙述都会被当作注释
(以上面为例，And子句将被SQL视为注释语句)

```
mysql> select * from userinfo where id='Admin'-- and pwd='12345';  
-> ;  
+-----+-----+  
| id    | pwd  |  
+-----+-----+  
| Admin | 123  |  
+-----+-----+  
1 row in set (0.00 sec)
```

`select * from table where name= ‘ “+un+” ’`

假设un输入: 12345' or 1=1

`select * from table where name='12345' or 1=1`

假设un输入:xxx; DROP TABLE SiteUsers WHERE 1=1

`select * from table`

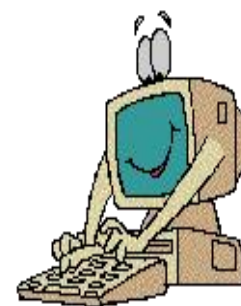
`where name=xxx; DROP TABLE SiteUsers WHERE 1=1`

思考：如何进行SQL注入的预防？



➤ 用户标识与鉴定 (Identification & Authentication)

- 静态口令
- 动态口令
- 生物特征（指纹、刷脸、虹膜等）
- 智能卡
- 入侵检测



➤ 存取控制

- 定义用户权限
- 合法权限检查



这两部分共同组成了DBMS的安全子系统

● 常用存取控制方

- 1) 自主存取控制 (Discretionary Access Control, 简称DAC)
C2级, 灵活
- 2) 强制存取控制 (Mandatory Access Control, 简称 MAC)
B1级, 严格

自主存取控制方法(DAC)

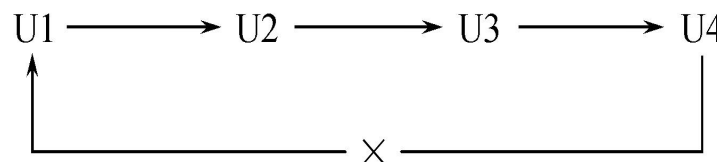
- 通过GRANT和REVOKE语句实现。
- 主要操作对象如下表：

对象类型	对象	操 作 类 型
数据库	模式	CREATE DATABASE
	基本表	CREATE TABLE, ALTER TABLE
模式	视图	CREATE VIEW
	索引	CREATE INDEX
数据	基本表 视图	SELECT, INSERT, UPDATE, DELETE, REFERENCES, ALL PRIVILEGES
	属性列	SELECT, INSERT, UPDATE, REFERENCES
		ALL PRIVILEGES

- 授权

- **GRANT** <权限>[, 权限]...
[ON <对象类型> <对象名>]
TO <用户>[, <用户>]...
[WITH GRANT OPTION]

不允许循环传播



1. DBA拥有数据库操作的所有权限，他可以将权限赋予其他用户。
2. 建立数据库对象用户称为该对象的属主 (OWNER)，他拥有该对象的所有操作权限。
3. 接受权限的用户可以是一个或多个具体用户，也可以是全体用户 (PUBLIC)。
4. **WITH GRANT OPTION**: 决定是否有传播权限的权利

4.2 数据库安全性控制 - DCL



```
create user 'user1'@'localhost' identified by '123456';
```

例1 把查询Student表权限授给用户U1

```
GRANT SELECT  
ON TABLE Student  
TO U1
```

例2 把对Student表和Course表的全部权限授予用户U2和U3

```
GRANT ALL PRIVILIGES  
ON TABLE Student, Course  
TO U2, U3
```

```
GRANT ALL  
ON TABLE Course  
TO U2, U3
```

```
GRANT ALL  
ON TABLE Student  
TO U2, U3
```

例3 把对表SC的查询权限授予所有用户

```
GRANT SELECT  
ON TABLE SC  
TO PUBLIC
```

MySQL不支持public关键字
(MySQL如何实现同等功能?)

例4 把查询Student表和修改学生学号的权限授给用户U4

```
GRANT UPDATE(Sno), SELECT  
ON TABLE Student  
TO U4
```

例5 把对表SC的INSERT权限授予U5用户，并允许他再将此权限授予其他用户

```
GRANT INSERT  
ON TABLE SC  
TO U5
```

WITH GRANT OPTION

-->U5--> U6--> U7

例6 DBA把在数据库S_C中建立表的权限授予用户U8

```
GRANT CREATETABLE  
ON DATABASE S_C  
TO U8
```

- 收回权限

- REVOKE语句的一般格式为:

REVOKE <权限>[, <权限>]...

[ON <对象类型> <对象名>]

FROM <用户>[, <用户>]...

[CASCADE/RESTRICT]

[例7] 把用户U4修改学生学号的权限收回。

```
REVOKE  UPDATE (Sno)
ON      TABLE   Student
FROM    U4;
```

- 收回权限

[例8] 收回所有用户对表SC的查询权限。

```
REVOKE    SELECT
ON         TABLE SC
FROM       PUBLIC;
```

MySQL不支持public关键字

[例9] 把用户U5对SC表的INSERT权限收回

```
REVOKE    INSERT
ON         TABLE SC
FROM       U5 ;
```

系统将收回直接或间接从U5处获得的对SC表的INSERT权限：

 -->U5--> U6--> U7

收回U5、U6、U7获得的对SC表的INSERT权限：

 <--U5<-- U6<-- U7

- MySQL中的权限相关概念

- 用户名 (user) : 访问数据库用户名
- 角色 (role) : 用户权限的组合 (主体集合)

相关SQL语句:

```
create user u1 identified by '123';  
create role r1;  
grant select on student.* to 'r1'@'%';  
grant 'r1'@'%' to 'u1'@'%';  
flush privileges ;
```

1. MySQL设置权限后需要刷新:
flush privileges ;

2. ROLE需要激活后权限才生效:
SET DEFAULT ROLE命令
或者
SET global
activate_all_roles_on_login=ON

查看所有用户: select user,host from mysql.user;
查看某个用户权限: show grants for 'user1'@'localhost'

■ 在MySQL中的应用:

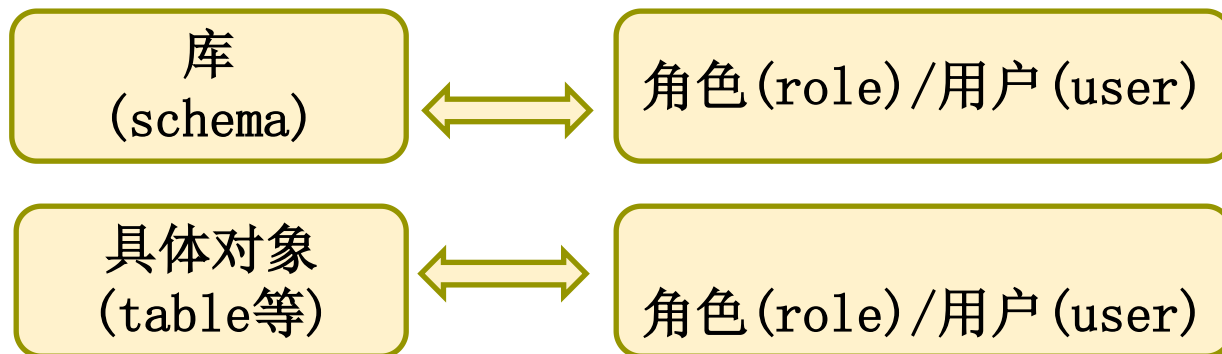
例1: 对于表s: 授予角色r2具有所有权限, u1具有查询、插入和修改学生系别的权限, u1的查询权限可以传播。
给角色r1授予对所有表的查询权限

```
grant all on s to r2;  
grant update on s to 'u1'@'%' with grant option;  
grant update on s to 'u1'@'localhost' with grant option;  
grant update on s to 'u1'@'192.168.1.108' with grant option;  
grant insert, update(sdept) on s to u1;  
grant select on *.* to r1;
```

例2: 对于表s: 收回用户r2的所有权限, 收回u1的查询权限。

```
revoke all on s from r2  
revoke select on student from u1@%
```

□ 权限的设定有多个入口



□ 当在不同入口设定的权限**不冲突**时，最终权限为**权限叠加**

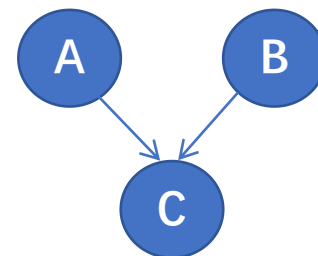
□ 当在不同入口设定的权限**冲突**时，最终的权限为**否定的优先**

（如：对某张表，user1具有**选择权限**，而role1**拒绝选择权限**，user1属于role1，则user1**最终为拒绝选择**）

关于以下Revoke的操作结果是什么？

A授权限X给C，B授相同权限X给C；

当A从C回收权限X后，C是否还有权限X？

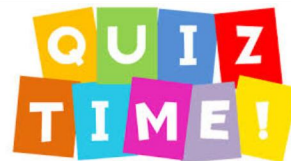


- ☐ A C没有权限X
- ☐ B C仍然具有B赋予的权限X
- ☒ C 不确定

DBMS如何实现权限管理的功能呢？

提交

Quiz Time



```
mysql> select * from columns_priv limit 1;  
Empty set (0.00 sec)
```

```
mysql> grant select, update(grade) on student.sc to 'u2'@'%';  
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> select * from tables_priv limit 1;
```

Host	Db	User	Table_name	Grantor	Timestamp	Table_priv	Column_priv
%	student	u2	sc	root@localhost	0000-00-00 00:00:00	Select	Update

```
1 row in set (0.00 sec)
```

```
mysql> select * from columns_priv limit 1;
```

Host	Db	User	Table_name	Column_name	Timestamp	Column_priv
%	student	u2	sc	grade	0000-00-00 00:00:00	Update

```
1 row in set (0.00 sec)
```

```
mysql> revoke select, update(grade) on student.sc from 'u2'@'%';  
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> select * from columns_priv limit 1;  
Empty set (0.00 sec)
```

```
mysql> select * from tables_priv limit 1;
```

Host	Db	User	Table_name	Grantor	Timestamp	Table_priv	Column_priv
localhost	mysql	mysql.session	user	boot@	0000-00-00 00:00:00	Select	

```
1 row in set (0.00 sec)
```

OceanBase的Grant语法

```
all_tenant_ots_user_level_history
all_tenant_profile
all_tenant_profile_history
all_tenant_role_grantee_map
all_tenant_role_grantee_map_history
all_tenant_scheduler_job
all_tenant_scheduler_job_run_detail
all_tenant_scheduler_program
all_tenant_scheduler_program_argument
all_tenant_security_audit
all_tenant_security_audit_history
all_tenant_security_audit_record
all_tenant_sysauth
all_tenant_sysauth_history
all_tenant_tablespace
all_tenant_tablespace_history
all_tenant_time_zone
all_tenant_time_zone_name
all_tenant_time_zone_transition
all_tenant_time_zone_transition_type
all_tenant_trigger
all_tenant_trigger_history
all_tenant_user_failed_login_stat
all_type
all_type_attr
all_type_attr_history
all_type_history
all_unit
all_unit_config
all_user
all_user_history
all_weak_read_service
all_zone
all_zone_merge_info
tenant_parameter
t1
t2
+-----+
450 rows in set (0.003 sec)

MySQL [oceanbase]> show tables;
```

社区V4.0版：450
个系统表（视图）

```
GRANT priv_type
ON priv_level
TO user_specification [, user_specification]...
[WITH with_option ...]
```

```
priv_type:
| ALTER
| CREATE
| CREATE USER
| CREATE VIEW
| DELETE
| DROP
| GRANT OPTION
| INDEX
| INSERT
| PROCESS
| SELECT
| SHOW DATABASES
| SHOW VIEW
| SUPER
| UPDATE
| USAGE
```

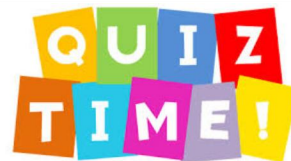
```
priv_level:
| *
| *.*
| database_name.*
| database_name.table_name
| table_name
| database_name.routine_name
```

```
user_specification:
user [IDENTIFIED BY [PASSWORD] 'password']
```

```
with_option:
GRANT OPTION
```

<https://www.oceanbase.com/docs/common-oceanbase-database-cn-100000000001579167>

Quiz Time



```
MySQL [oceanbase]>
MySQL [oceanbase]>
MySQL [oceanbase]> grant update on testln.sc to obu1;
Query OK, 0 rows affected (0.022 sec)

MySQL [oceanbase]> revoke update on testln.sc from obu1;
Query OK, 0 rows affected (0.019 sec)

MySQL [oceanbase]> grant update(grade) on testln.sc to obu1;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'update(grade) on testln.sc to obu1' at line 1
MySQL [oceanbase]>
```

```
MySQL [oceanbase]> select * from __all_table_privilege\G
***** 1. row *****
gmt_create: 2023-09-23 15:50:12.297285
gmt_modified: 2023-09-23 15:50:12.297285
tenant_id: 0
user_id: 502929
database_name: testln
table_name: sc
priv_alter: 0
priv_create: 0
priv_delete: 0
priv_drop: 0
priv_grant_option: 0
priv_insert: 0
priv_update: 1
priv_select: 1
priv_index: 0
priv_create_view: 0
priv_show_view: 0
1 row in set (0.000 sec)
```

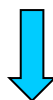
OB如何实现对
列控制权限呢？

```
MySQL [oceanbase]> select user_id, user_name from __all_user;
+-----+-----+
| user_id | user_name |
+-----+-----+
| 502929 | obu1      |
| 200005 | ORAAUDITOR |
| 200001 | root      |
+-----+-----+
3 rows in set (0.000 sec)

MySQL [oceanbase]>
```

- 自主存取控制 (DAC) 的缺点

- 可能存在数据的“无意泄露”
- 原因：这种机制仅仅通过对数据的存取**权限**来进行安全控制，而数据本身并无安全性标记
- 解决：对系统控制下的所有**主客体**实施强制存取控制策略



MAC

- 强制存取控制 (MAC)

- 1) **主体**是系统中的活动实体：
DBMS所管理的实际用户/代表用户的各进程
- 2) **客体**是系统中的被动实体，是受主体操纵的
文件/ 基本表/ 索引/ 视图

➤ 强制存取控制 (MAC)

- (1) 每一个数据对象被标以一定的密级
- (2) 每一个用户也被授予某一个级别的许可证
- (3) 对于任意一个对象，只有具有合法许可证的用户才可以存取

● 敏感度标记 (Level)

- 绝密 (Top Secret)
- 机密 (Secret)
- 可信 (Confidential)
- 公开 (Public)

主体敏感度标记：许可证级别
客体敏感度标记：客体的密级

是对数据本身进行密级标记，无论数据如何复制，标记和数据是一个不可分的整体

目前：DB2, Oracle (基于安全标签的访问控制)

MAC控制规则

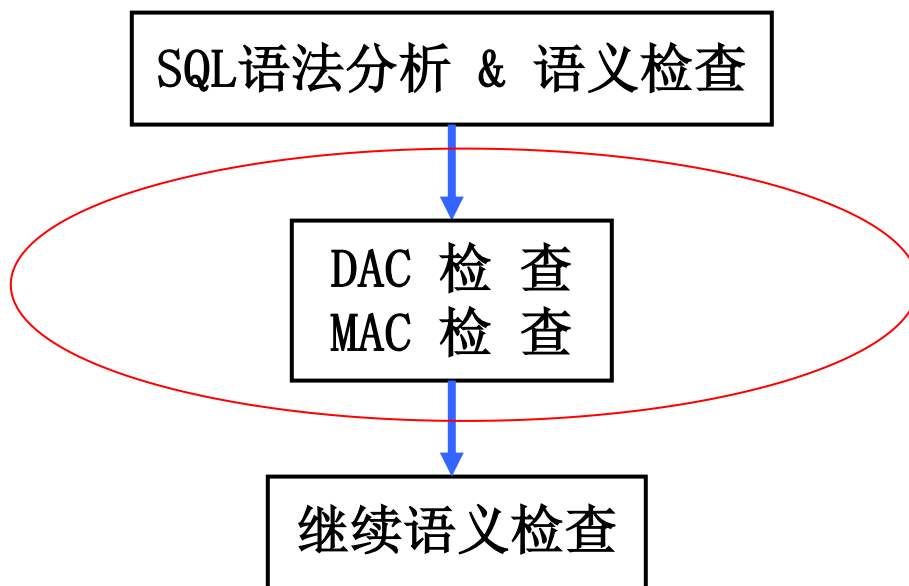
- (1) 仅当主体的许可证级别**大于或等于**客体的密级时，该主体才能**读**取相应的客体
- (2) 仅当主体的许可证级别**等于**客体的密级时，该主体才能**写**相应的客体

- MAC（2）的修正规则

主体的许可证级别 \leq 客体的密级 \rightarrow 主体能写客体

共同点：禁止了拥有**高许可证级别**的主体**更新低密级的数据对象**

DBMS的安全机制：DAC+MAC



- ❖ 先进行DAC检查，通过DAC检查的数据对象再由系统进行MAC检查，只有通过MAC检查的数据对象方可存取。

4.1.计算机安全性概述

4.2.数据库安全性控制

4.3.视图机制

4.4.审计(Audit)

4.5.数据加密

4.6.其他安全性保护



视图的安全保护

把要保密的数据对无权存取这些数据的用户隐藏起来，对数据提供一定程度的安全保护。

- 主要功能是提供数据**独立性**，其安全保护功能不够精细，往往远不能达到应用系统的要求。
- 间接实现了支持存取谓词的用户权限定义。

[例]建立主修计算机专业学生的视图，把对该视图的SELECT权限授予王平，把该视图上的所有操作权限授予张明

1) 先建立计算机专业学生的视图CS_Student

```
CREATE VIEW CS_Student AS  
SELECT Sno, Sname  
FROM Student  
WHERE Smajor= 'CS' ;
```

2) 在视图上进一步定义存取权限

```
GRANT SELECT  
ON CS_Student  
TO 王平 ;  
GRANT ALL PRIVILIGES  
ON CS_Student  
TO 张明;
```

4.1.计算机安全性概述

4.2.数据库安全性控制

4.3.视图机制

4.4.审计(Audit)

4.5.数据加密

4.6.其他安全性保护



● 什么是审计

- 启用一个专用的审计 **日志** (Audit Log)
用途：将用户对数据库的**所有操作记录**在上面
- DBA可以利用审计日志中的追踪信息找出非法存取数据的人
- **C2以上**安全级别的DBMS**必须**具有审计功能

● 审计功能的可选性

- 审计很费时间和空间
- DBA可以根据应用对安全性的要求，灵活地打开或关闭审计功能。

● 审计功能的关键字

- AUDIT/NOAUDIT（教材提及）
 - 对表 t1 的所有 INSERT、UPDATE 和 DELETE 操作进行审计。

```
1. obclient> AUDIT INSERT, UPDATE, DELETE on t1;  
2. Query OK, 0 rows affected (31.62 sec)
```

- MySQL中的审计日志：社区版没有审计功能

审计日志例子：

```
[root@smiletest mysql]# tail -f server_audit.log
20190903 09:24:18,smiletest,root,localhost,2,0,FAILED_CONNECT,,,1045
20190903 09:24:23,smiletest,root,localhost,3,0,CONNECT,,,0
20190903 09:24:23,smiletest,root,localhost,3,3,QUERY,,'select @@version_comment limit 1',0
20190903 09:24:25,smiletest,root,localhost,3,4,QUERY,,'SHOW VARIABLES LIKE \'%audit%\'',0
20190903 09:24:45,smiletest,root,localhost,3,5,QUERY,,'show databases',0
20190903 09:25:12,smiletest,root,localhost,3,0,DISCONNECT,,,0
```

MySQL中的其他日志
(非审计日志)

- 1: 重做日志 (redo log)
- 2: 回滚日志 (undo log)
- 3: 二进制日志 (binlog)
- 4: 错误日志 (errorlog)
- 5: 慢查询日志 (slow query log)
- 6: 一般查询日志 (general log)
- 7: 中继日志 (relay log)。

4.1.计算机安全性概述

4.2.数据库安全性控制

4.3.视图机制

4.4.审计(Audit)

4.5.数据加密

4.6.其他安全性保护



● 数据加密（存储加密 + 传输加密）

- 防止数据库中数据在存储和传输中失密的有效手段

● 加密的基本思想

- 根据一定的算法将原始数据（术语为明文，Plain text）变换为不可直接识别的格式（术语为密文，Cipher text）
- 不知道解密算法的人无法获知数据的内容。

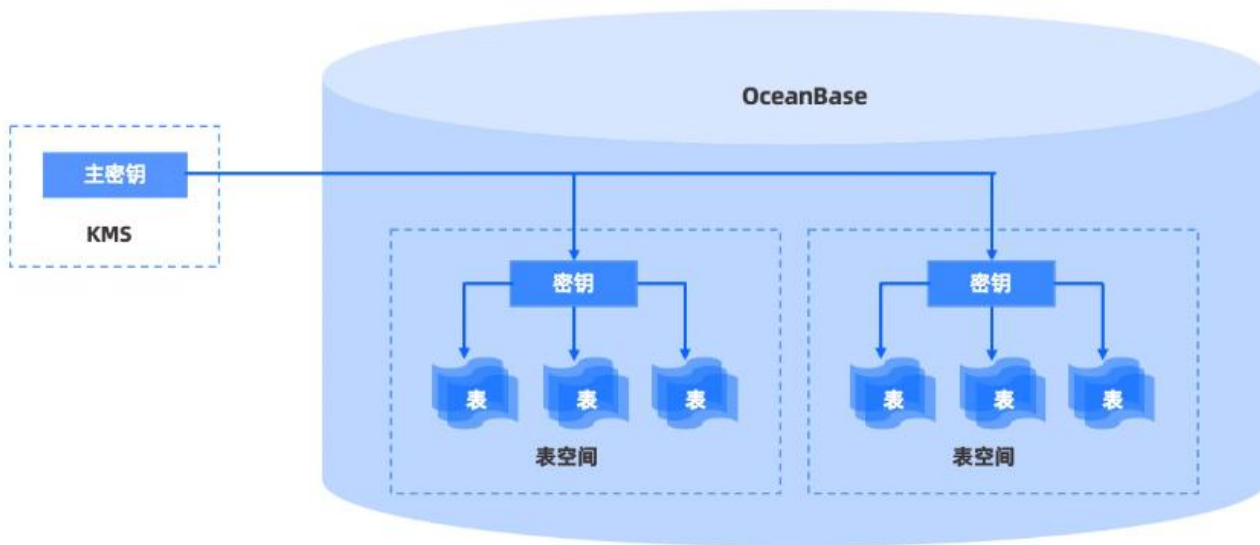
● 加密的基本方法

- **替换方法：**使用密钥（Encryption Key）将明文中的每一个字符转换为密文中的一个字符
- **置换方法：**将明文的字符按不同的顺序重新排列
- **混合方法：**美国1977年制定的官方加密标准：数据加密标准

4.5 数据加密 - 数据库OB内核



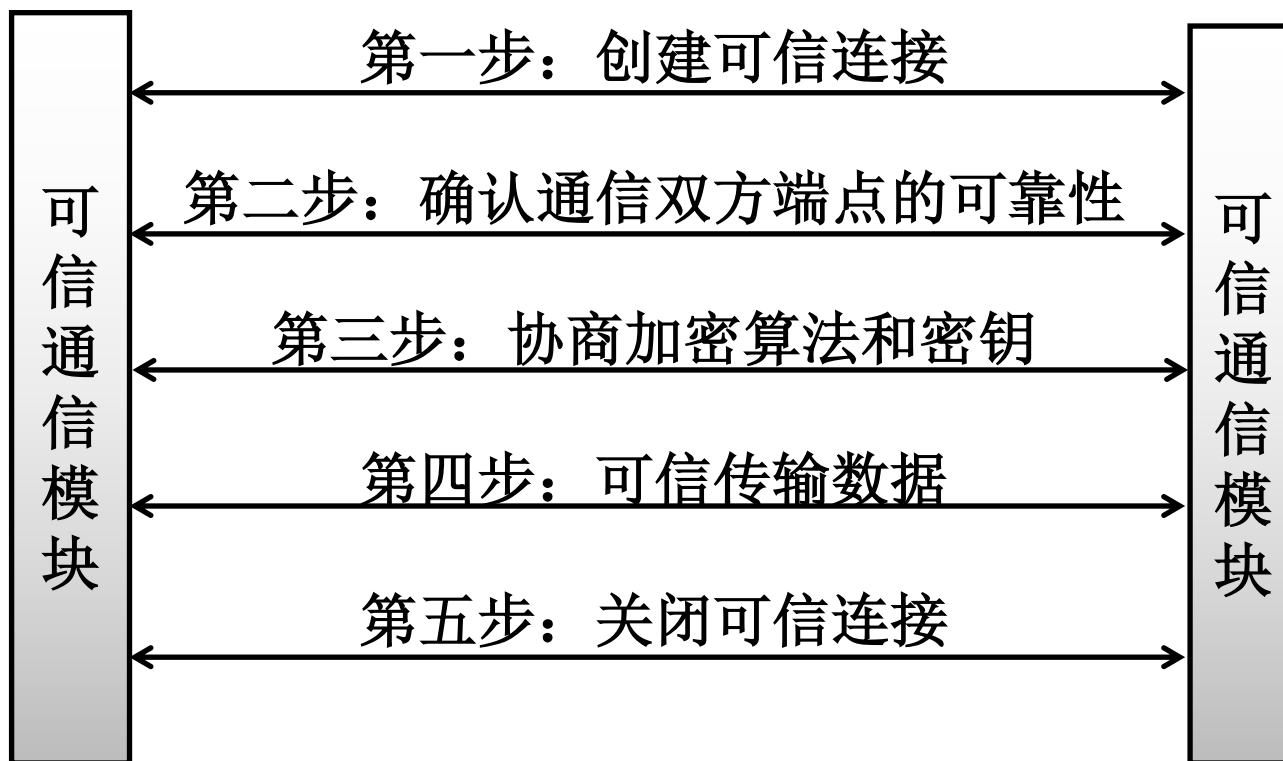
OceanBase中的数据加密 —— 存储加密



- 透明数据加密使用两级密钥体系实现加解密功能
- 开启加密的最小粒度为数据库中的一个表
- 需要开启加密的表需要放到一个加密的表空间（tablespace）中。

<https://www.oceanbase.com/docs/common-oceanbase-database-cn-1000000000033718>

4.5 数据加密 - 传输加密



基于 SSL protol的数据库管理系统可信传输示意图

4.5 数据加密 - MYSQL传输加密



正在捕获 本地连接 2

文件(F) 编辑(E) 视图(V) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

tcp.port == 24850

No.	Time	Source	Destination	Protocol	Length	Info
50	6.101540	192.168.16.76	192.168.16.115	TCP	66	64088 → 24850 [SYN] Seq=
51	6.101754	192.168.16.115	192.168.16.76	TCP	66	24850 → 64088 [SYN, ACK]
52	6.101794	192.168.16.76	192.168.16.115	TCP	54	64088 → 24850 [ACK] Seq=
53	6.102069	192.168.16.115	192.168.16.76	TCP	131	24850 → 64088 [PSH, ACK]
54	6.102121	192.168.16.76	192.168.16.115	TCP	139	64088 → 24850 [PSH, ACK]
55	6.102268	192.168.16.115	192.168.16.76	TCP	60	24850 → 64088 [ACK] Seq=
56	6.120070	192.168.16.115	192.168.16.76	TCP	65	24850 → 64088 [PSH, ACK]
57	6.120327	192.168.16.76	192.168.16.115	TCP	91	64088 → 24850 [PSH, ACK]
58	6.121858	192.168.16.115	192.168.16.76	TCP	132	24850 → 64088 [PSH, ACK]
59	6.323193	192.168.16.76	192.168.16.115	TCP	54	64088 → 24850 [ACK] Seq=
80	11.505585	192.168.16.76	192.168.16.115	TCP	86	64088 → 24850 [PSH, ACK]
81	11.507305	192.168.16.115	192.168.16.76	TCP	423	24850 → 64088 [PSH, ACK]
84	11.705986	192.168.16.76	192.168.16.115	TCP	54	64088 → 24850 [ACK] Seq=

Frame 50: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Ethernet II, Src: fe:fc:fe:73:66:7f (fe:fc:fe:73:66:7f), Dst: Dell_27:fc:01 (90:b1:1c:27:fc:01)
Internet Protocol Version 4, Src: 192.168.16.76, Dst: 192.168.16.115

```
0000  90 b1 1c 27 fc 01 fe fc fe 73 66 7f 08 00 45 00  ...sf...E-
0010  00 34 30 3d 40 00 80 06 00 00 c0 a8 10 4c c0 a8  ...40=@...L-
0020  10 73 fa 58 61 12 4c d6 31 ba 00 00 00 00 80 02  ...sXaL 1....
0030  20 00 a2 36 00 00 02 04 05 b4 01 03 03 08 01 01  ...6.....
0040  .....
```

管理员: C:\Windows\system32\cmd.exe - mysql -h 192.168.16.115

```
C:\Users\User>mysql -h 192.168.16.115 -P24850 -uadmin -padmi
mysql: [Warning] Using a password on the command line inter
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 29
Server version: 5.5.1 Vaobase

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/o
affiliates. Other names may be trademarks of their respectiv
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the curren

mysql> show variables like 'ssl%';
+-----+-----+
| variable_name | value |
+-----+-----+
| have_ssl      | NO    |
| ssl_capath    |       |
| ssl_cipher    | Not in use |
| have_openssl  | NO    |
| ssl_key       | ssl/server-key.pem |
| ssl_ca        | ssl/ca.pem |
| ssl_crl       |       |
| ssl_crlpath   |       |
| ssl_cert      | ssl/server-cert.pem |
+-----+-----+
9 rows in set (0.00 sec)

mysql>
```

MYSQL 未开启SSL连接（普通连接模式）

4.5 数据加密 - MYSQL传输加密



```
[root@k8s-master ca]# mysql -u usessl --ssl-ca=ca.pem --ssl-cert=client-cert.pem --ssl-key=client-key.pem -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.31 MySQL Community Server - GPL

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

You are enforcing ssl connection via unix socket. Please consider
switching ssl off as it does not make connection via unix socket
any more secure.
mysql>
```

The image shows a Wireshark packet capture of a MySQL connection. The main pane displays a list of packets, with the SSL handshake (TLSv1.3) highlighted. The 'Info' pane on the right shows the details of the selected packet, including the 'Client Hello' and 'Server Hello' messages. The 'SSL cipher' is listed as 'TLS_AES_256_GCM_SHA384'. The 'MySQL' pane on the right shows the output of the 'show variables like 'ssl';' command, which lists various SSL-related variables, including 'have_ssl', 'ssl_cipher', 'ssl_key', 'ssl_ca', 'ssl_cert', etc.

No.	Time	Source	Destination	Protocol	Length	Info
18	2.809706	192.168.16.76	192.168.16.115	TCP	66	64090 → 24850 [SYN] Seq=
19	2.809881	192.168.16.115	192.168.16.76	TCP	66	24850 → 64090 [SYN, ACK] Seq=
20	2.809926	192.168.16.76	192.168.16.115	TCP	54	64090 → 24850 [ACK] Seq=
21	2.810242	192.168.16.115	192.168.16.76	TLSv1.3	131	Continuation Data
22	2.810280	192.168.16.76	192.168.16.115	TLSv1.3	90	Continuation Data
23	2.810400	192.168.16.115	192.168.16.76	TCP	60	24850 → 64090 [ACK] Seq=
24	2.811267	192.168.16.76	192.168.16.115	TLSv1.3	353	Client Hello
25	2.811393	192.168.16.115	192.168.16.76	TCP	60	24850 → 64090 [ACK] Seq=
26	2.815344	192.168.16.115	192.168.16.76	TLSv1.3	1514	Server Hello, Change Cip
27	2.815379	192.168.16.115	192.168.16.76	TLSv1.3	853	Application Data, Applic
28	2.815392	192.168.16.76	192.168.16.115	TCP	54	64090 → 24850 [ACK] Seq=
29	2.816050	192.168.16.76	192.168.16.115	TLSv1.3	164	Change Cipher Spec, Appl
30	2.816118	192.168.16.76	192.168.16.115	TLSv1.3	161	Application Data
31	2.816386	192.168.16.115	192.168.16.76	TLSv1.3	293	Application Data
32	2.816573	192.168.16.115	192.168.16.76	TLSv1.3	293	Application Data

Frame 36: 154 bytes on wire (1232 bits), 154 bytes captured (1232 bits) on interface 0
Ethernet II, Src: Dell_27:fc:01 (90:b1:1c:27:fc:01), Dst: fe:fc:fe:73:66:7f (fe:fc:fe:73:66:7f)
Internet Protocol Version 4, Src: 192.168.16.115, Dst: 192.168.16.76

0000 fe fc fe 73 66 7f 90 b1 1c 27 fc 01 08 00 45 00 ...sf....E
0010 00 8c e3 04 40 00 40 06 b5 57 c0 a8 10 73 c0 a8 ...@@-W-s
0020 10 4c 61 12 fa 5a 6b fa 72 69 84 07 47 5e 50 18 ...La-Zk-ri-G^P
0030 00 53 5e e7 00 00 17 03 03 00 5f 5e 69 6e b0 1c ...S^.....^in
0040 bd 66 4b 7c 52 96 34 90 aa af 61 bf ab 50 24 cf ...fk|R-4...a-PS
0050 eb 85 c5 97 b3 bb 96 81 1f 88 f6 31 f8 e4 c4 a8 ...1
0060 98 12 3e aa cf 95 6b a7 f0 79 8e 26 5c d2 cb 26 ...->...k-y-&-&

mysql> show variables like 'ssl';

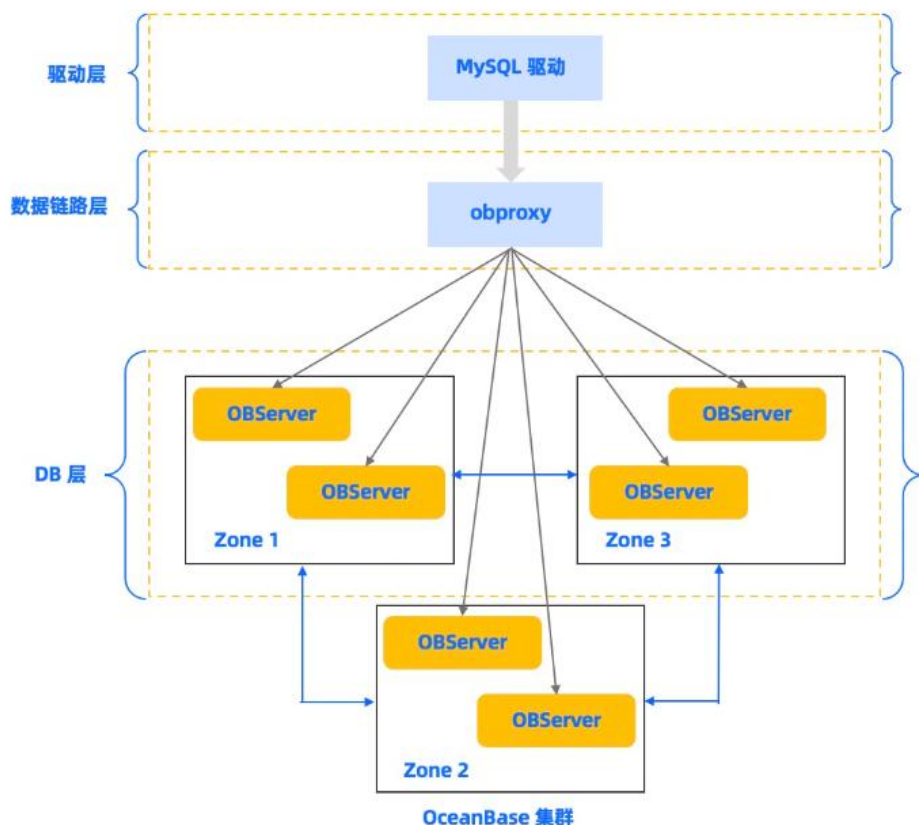
variable_name	value
have_ssl	YES
ssl_capath	
ssl_cipher	TLS_AES_256_GCM_SHA384
have_openssl	YES
ssl_key	ssl/server-key.pem
ssl_ca	ssl/ca.pem
ssl_crl	
ssl_crlpath	
ssl_cert	ssl/server-cert.pem

9 rows in set (0.00 sec)

MYSQL 开启SSL连接（安全加密模式）

OceanBase中的数据加密 —— 传输加密

- **安全传输层协议（TLS）** 用于在两个通信应用程序之间提供保密性和数据完整性。



- **MySQL 协议（扩展版）：**
 - 驱动层 \leftrightarrow 数据链路层
 - 数据链路层 \leftrightarrow DB 层
- **OB-RPC 协议（OB自有）：**
 - 节点与节点之间
 - 节点与 liboblog、ob_admin 等之间

<https://www.oceanbase.com/docs/common-oceanbase-database-cn-1000000000033720>

4.5 数据加密 - 数据库OB内核



OceanBase中的数据加密 —— 传输加密

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000			TCP	74	25424 → 30883 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=279...
2	0.000016	182.119.	182.119.	TCP	74	30883 → 25424 [SYN, ACK] Seq=0 Ack=1 Win=43690 Len=0 MSS=65495 SACK_PERM=...
3	0.000029	182.119.	182.119.	TCP	66	25424 → 30883 [ACK] Seq=1 Ack=1 Win=44032 Len=0 TSval=2798694352 TSecr=27...
4	0.000115	182.119.	182.119.	TLSv1.2	144	Ignored Unknown Record
5	0.000138	182.119.	182.119.	TCP	66	25424 → 30883 [ACK] Seq=1 Ack=79 Win=44032 Len=0 TSval=2798694352 TSecr=2...
6	0.001096	182.119.	182.119.	TLSv1.2	102	Ignored Unknown Record
7	0.001104	182.119.	182.119.	TCP	66	30883 → 25424 [ACK] Seq=79 Ack=37 Win=44032 Len=0 TSval=2798694353 TSecr=...
8	0.003235	182.119.	182.119.	TLSv1.2	165	Client Hello
9	0.003244	182.119.	182.119.	TCP	66	30883 → 25424 [ACK] Seq=79 Ack=136 Win=44032 Len=0 TSval=2798694355 TSecr=...
10	0.009859	182.119.	182.119.	TLSv1.2	2771	Server Hello, Certificate, Server Key Exchange, Certificate Request, Serv...
11	0.009885	182.119.	182.119.	TCP	66	25424 → 30883 [ACK] Seq=136 Ack=2784 Win=175104 Len=0 TSval=2798694362 TS...
12	0.020688	182.119.	182.119.	TLSv1.2	2376	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, ...
13	0.020722	182.119.	182.119.	TCP	66	30883 → 25424 [ACK] Seq=2784 Ack=2446 Win=175104 Len=0 TSval=2798694373 T...
14	0.026406	182.119.	182.119.	TLSv1.2	1108	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
15	0.026487	182.119.	182.119.	TLSv1.2	180	Application Data
16	0.027305	182.119.	182.119.	TLSv1.2	106	Application Data
17	0.027492	182.119.	182.119.	TLSv1.2	132	Application Data
18	0.028071	182.119.	182.119.	TLSv1.2	278	Application Data
19	0.067290	182.119.	182.119.	TCP	66	25424 → 30883 [ACK] Seq=2626 Ack=4078 Win=185856 Len=0 TSval=2798694420 T...
20	1.067293	182.119.	182.119.	TCP	66	[TCP Keep-Alive] 30883 → 25424 [ACK] Seq=4077 Ack=2626 Win=175104 Len=0 T...
21	1.067304	182.119.	182.119.	TCP	66	[TCP Keep-Alive ACK] 25424 → 30883 [ACK] Seq=2626 Ack=4078 Win=185856 Len=...
22	4.398162	182.119.	182.119.	TLSv1.2	133	Application Data
23	4.398405	182.119.	182.119.	TLSv1.2	212	Application Data
24	4.398432	182.119.	182.119.	TCP	66	25424 → 30883 [ACK] Seq=2693 Ack=4224 Win=190976 Len=0 TSval=2798698751 T...

SSL握手

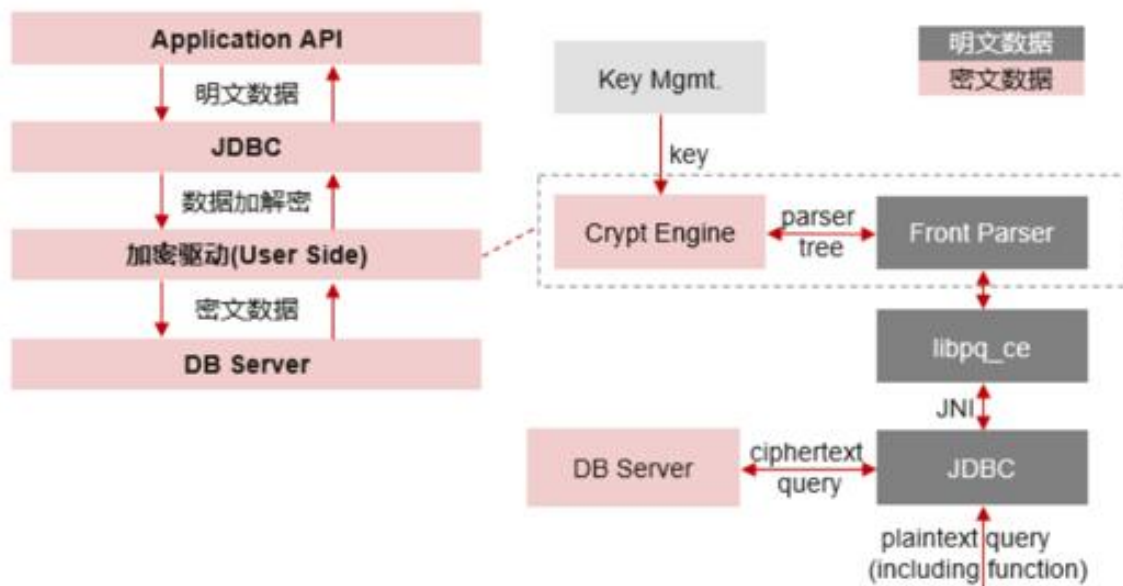
- Handshake Protocol: Client Key Exchange
- TLSv1.2 Record Layer: Handshake Protocol: Certificate Verify
 - Content Type: Handshake (22)
 - Version: TLS 1.2 (0x0303)
 - Length: 264
- Handshake Protocol: Certificate Verify
- TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
 - Content Type: Change Cipher Spec (20)
 - Version: TLS 1.2 (0x0303)
 - Length: 1

tcpdump工具抓包验证

4.5 数据加密 - 华为全密态数据库



openGauss全密态数据库



核心：客户端解析用户全部的输入输出语句，识别出已定义的敏感数据，并且进行自动化加解密。用户使用全密态数据库时，从输入语法，客户端发送到服务端，服务端执行并将结果返回给客户端的整个流程中，用户唯一能感知到的只有输入语法以及获得返回结果这两步。

https://blog.csdn.net/GaussDB/article/details/120843218?spm=1001.2101.3001.6650.7&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromBaidu%7ERate-7-120843218-blog-121101435.pc_relevant_multi_platform_whitelistv6&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromBaidu%7ERate-7-120843218-blog-121101435.pc_relevant_multi_platform_whitelistv6&utm_relevant_index=8

4.1.计算机安全性概述

4.2.数据库安全性控制

4.3.视图机制

4.4.审计(Audit)

4.5.数据加密

4.6.其他安全性保护



- 推理控制机制（统计数据安全）

- 基于函数依赖的推理
- 基于敏感关联的推理

- 隐蔽信道

经过隐蔽信息通道（如SQL执行后的反馈信息）获取信息

- 数据隐私(data privacy)

- 控制不愿或不便他人知道的个人数据的能力
- 范围很广：数据收集、数据存储、数据处理和数据发布等各阶段

- “三权分立”的安全管理机制

解决数据库管理员权限过于集中的问题，遵照GB/T20273-2019，引进“三权分立”的安全管理机制

重点

- 数据库安全性实现的方法
- 自主存取控制（DAC）与强制存取控制（MAC）
- 授权（Authorization）与回收（GRANT&REVOKE）
- 数据库角色

难点

- 强制存取控制（MAC）的理解



作业： 实验预习： 第4章： 6, 7, 8。
(不用书面提交，实验预习)

