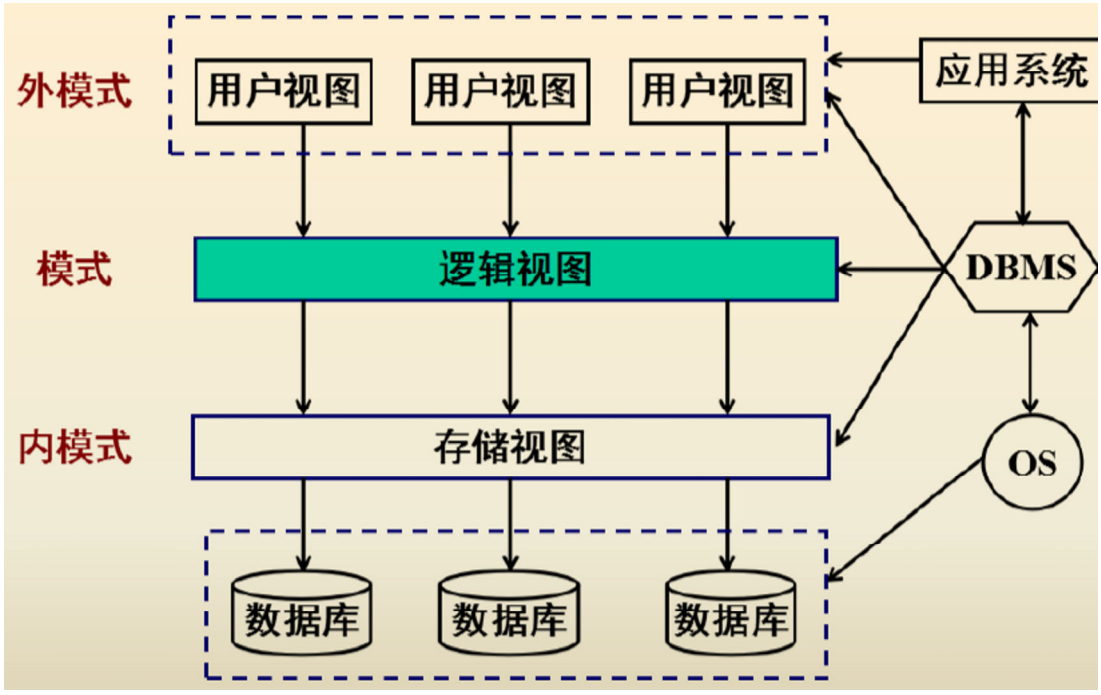
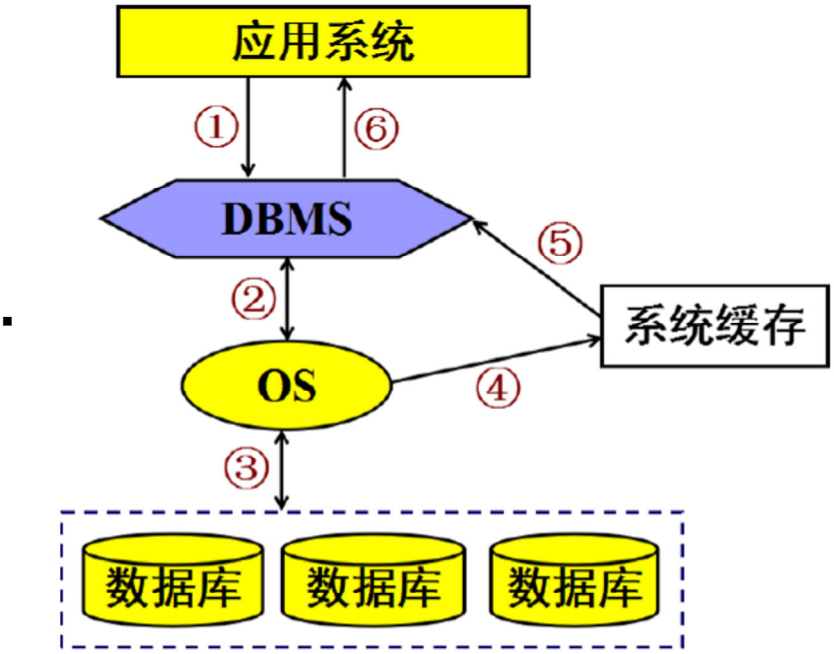
7 数据库安全

2019年4月10日　　23:37

1 数据库安全概述

- 数据库
  - 按照数据结构来组织、存储和管理数据的仓库。
  - 数据库的层次结构
    - 

- 数据库管理系统（DBMS）
  - 用于建立、使用和管理数据库的软件系统。如:Oracle、SQLServer、MySQL等
  - 组成：主要包括存储管理器和查询处理器
  - 功能：
    - 建立、管理和维护数据库；
    - 为用户及程序提供数据访问；
    - 保证数据库的安全性。
  - 数据库系统的工作流程



2 数据库的安全需求

- 完整性：数据的正确性和相容性
  - 物理数据库完整性
    - 整个数据库损毁
    - 保证数据能够物理读取
  - 逻辑数据库完整性
    - 保护数据库的结构
  - 元素完整性
    - 数据元素只能由授权用户改变
  - 防止对DBMS的非法访问和修改
  - 保护存储的数据、文件的安全性
- 机密性
  - 数据值
  - 数据值范围
  - 否定的查询结果
  - 可能的取值
- 可审计性
  - 维护完整性，受破坏后可恢复
  - 用户累加式访问受保护数据
  - 后台记录用户访问记录:推理用户意图
- 可用性
- 一致性
- 用户认证

完整性保护实例——两阶段更新

- 事例
  - 数据库中包含公司的办公用品使用和支出信息：纸、笔、夹子等。
  - 一个部门申请50盒夹子，仓库中存有107盒，如果仓库存量少于100盒则需要外购。
- 业务处理过程
  a. 仓库人员检查数据库中是否能够支付50盒夹子，如果不够支付，则拒绝部门申请，交易结束；
  b. 如果库存超过申请数量，仓库人员从库存中减去50，数据库中夹子数量107-50=57；
  c. 仓库人员准备支付物品：提供50盒夹子；
  d. 仓库人员检查库存量是否低于库存最小量，低则进行外购；57<100,生成采购单，在数据库中修改标志位为“onorder”；
  e. 支付实物 50盒夹子。
- 两阶段更新——准备阶段
  a. 检查数据库中的COMMITFLAG。如果值为1，不能执行操作，中止或等待，检查COMMITFLAG直到值为0；
  b. 比较库存夹子数量是否超过申请数量，如果少于申请数量则中止；
  c. 计算：TCLIPS=ONHANDREQUISITION
  d. 计算部门开销的办公费用BUDGET
  TBUDGET=BUDGETCOST，其中COST是50盒夹子的价钱；
  e. 检查TCLIPS是否低于最小库存量；低于最小库存量，设置TREORDER=TRUE;否则设置TREORDER=FALSE。
- 两阶段更新——提交阶段
  a. 置COMMITFLAG=1；
  b. 拷贝TCLIPS到数据库中的CLIPS；
  c. 拷贝TBUDGET到数据库中的BUDGET；
  d. 拷贝TREORDER到数据库中的REORDER；
  e. 通知申请部门领取物品，记录完成交易日志；
  f. 修改COMMITFLAG=0。

数据库安全威胁源分类

- 自然灾害
  - 由不可抗力造成的意外事故或灾难。
- 设备故障
  - 数据库应用系统或者数据库自身软硬件中的故障。
- 人为疏忽
  - 由授权用户造成的无意损害。
- 恶意攻击
  - 恶意的数据库开发、使用和管理人员所实施的攻击
- 管理漏洞

    - 安全管理制度缺失或者执行不力而导致安全问题。
- 数据库系统安全威胁80%来自内部用户的误操作和恶意操作。

(了解)数据库系统的安全防护

- 网络环境层次
    - 数据库的安全首先依赖于网络系统，网络系统的安全是数据库安全的第一道屏障。
- 宿主操作系统层次
    - 防止对DBMS的非法访问和修改
    - 保护存储的数据、文件的安全性
    - 对数据库用户进行系统登录认证
- 数据库管理系统层次
    - 功能
        - 用户/角色管理
        - 用户/角色授权
        - 身份认证
        - 特权管理
        - 操作审计
    - 保护数据的机密性
        - 访问控制：DAC，MAC，RBAC。
        - 加密
    - 保护数据的完整性
        - 访问控制：DAC，MAC，RBAC。
        - 认证
    - 保护数据的一致性
    - 保护系统的可用性
        - 备份/恢复
        - 资源使用限制
    - 并发控制
- 数据库应用系统层次
    - 用户/角色管理
    - 用户管理
    - 身份认证
    - 访问控制
    - 业务审计
    - 输入检查

3 数据库的安全机制

- 3.3 推理控制
- 身份认证
- 访问控制
- 安全审计

3.1 视图机制

- 原理：通过定义不同的视图，将用户无权访问的数据隐藏起来
- 实例：
  - 数据库中有如下用户信息表：Inf_tab:name,age,sex,job,sal,addr,phone
  - 现要求：用户Bob只能查询上表中的name,job,phone。可用视图机制实现。
  - 创建视图

```
CREATE VIEW inf_viewAS
SELECT name,job,phone
FROM inf_tab
```

  - 授予Bob上述视图的查询权限

```
GRANT SELECT ON inf_view TO Bob
```

3.2 数据库加密技术

- 库外加密
  - 将数据库的文件作为加密对象
    - 加 / 解密过程发生在DBMS之外，DBMS管理的是密文。
    - 加 / 解密过程大多在客户端实现，也可由专门的加密服务器或硬件完成
  - 优点：
    - 对DBMS的要求少
  - 缺点：
    - 效率低
    - 数据解密需要很大的时间和空间代价
- 库内加密
  - 加密对象为数据库中存储的数据，比如表、记录、字段等。
    - 库内加密在DBMS内核层实现加密。
    - 加 / 解密过程对用户与应用透明，数据在物理存取之前完成加 / 解密工作
  - 优点：
    - 加密的粒度可细化
    - 效率较高
  - 缺点：
    - 针对性较强
- 硬件加密
  - 在物理存储器与数据库系统之间的硬件加密装置
  - 加密解密操作由硬件中间设备完成
  - 缺点：

        - 硬件之间的兼容问题
        - 系统变得更复杂
- 特殊问题
    - 不能以整个数据库文件为单位进行加密
        - 脱密操作无法从文件中间开始
        - 由于数据共享需要，密钥管理困难
    - 部分字段不能加密
        - DBMS必须能够识别条件字段
        - 索引字段不能加密。
        - 表间的连接码字段不能加密

3.3 推理控制

- 利用数据之间的相互关系，从合法获得的低安全等级的数据中，推导出数据库中受高安全等级保护的内容，从而造成敏感信息泄露。
- 基于查询敏感数据的推理.例:

  P为公开数据表，它有两个属性：职员名和项目名

  S为涉密数据表，它有两个属性：项目名和项目类型

  如下查询可导致泄密：

    - ```
      select 职员名
      from P
      where P.项目名=S.项目名
      ```

(不考)4 SQLServer数据库安全

- 操作系统级的安全防线
    - OS身份认证和访问控制
- 服务器级的安全防线
    - SQLServer身份认证
- 数据库级的安全防线
    - 特定数据库自己的用户账户和角色
- 数据库对象级的安全防线
    - 用户必须在自己的权限范围内操作数据

SQLServer的身份认证模式

- Windows身份认证模式
    - 只要用户能够通过Windows系统的身份认证,即可连接到SQL Server服务器上
      ```

    - 前提：必须先由数据库管理员在SQL Server中创建与Windows账号对应的SQL Server账号。
- 混合身份认证模式
    - 用户在与SQL Server服务器连接时，既可以使用Windows身份认证，也可以使用SQL Server身份认证
    - 对于可信的Windows用户，直接采用Windows身份认证模式，否则SQL Server自行进行认证

SQLServer的账号安全性检查

- 检查无用的数据库账号
- 检查弱口令账号

  SQLCracker.exe

  forceSQL.exe
- 检查空口令账号

  Select name from syslogins where password is null

SQLServer的权限管理

- 语句权限
    - 创建对象的权限
    -

动作	对象	语句
备份	数据	backup database
	日志	backup log
创建	数据库	create database
	数据表	create table
	视图	create view
	存储过程	create procedure
	自定义函数	create function
	规则	create rule
	默认	create default

- 对象权限
    - 操作对象的权限

-

对象	操作
表或视图	select、insert、update 、delete
存储过程	execute
内嵌表值函数	select
表或视图的列	select和update

- 隐含权限
  - 通过角色传递得到的权限
  - 隐含权限是将用户加入角色，系统自动将角色的权限传递给成员的权限
- 相关命令
  - 授予权限—GRANT

    例：GRANT CREATE DATABASE TO user1，user2
  - 撤销权限—REVOKE

    例：REVOKE CREATE DATABASE FROM user1，user2
  - 禁止权限—DENY
    - 删除原有的权限，并禁止该用户或角色从其他角色继承禁止的权限

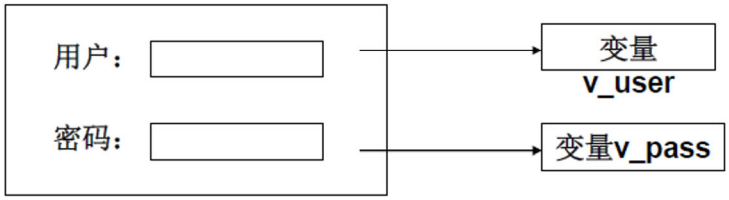
      例：禁止user创建表: DENY CREATE TABLE TO user

      例：禁止user对表readers进行DELETE操作: DENY DELETE ON readers TO user

SQLServer的入侵检测

- 系统日志
  - 数据库服务停止或重新启动
  - 执行扩展存储过程纪录
- 数据库管理系统日志
  - 多次登录失败纪录
  - 数据库登录日志
- 数据库应用日志
  - 执行插入的SQL语句
  - 检查用户输入的符号是否对系统有威胁

SQL注入攻击



- select * from user where user_name='v_user' and password='v_pass';
  +
  v_user=admin v_pass=1' or '1'='1
  =
  select * from user where user_name='admin' and password='1' or '1'='1' ;
- 最基本的防范
  - 不允许输入 '
  - 将输入的 ' 转换

SQLServer的扩展存储过程

- 存储过程
  SQL服务器上的一组预编译好的可以完成特定功能的SQL语句集，可分为两类：
  - 系统存储过程：由系统自动创建，完成的功能主要是从系统表中获取信息。
  - 自定义的存储过程：由用户为完成某一特定功能而编写的存储过程。
- 执行用法
  - 指定存储过程的名字，并给出其参数。
- xp_cmdshell可以直接执行系统命令
  - xp_cmdshell “dir c:\”
  - xp_cmdshell “net user TEST/add”
- 其他重要扩展存储过程
  - xp_regdeletekey、xp_regdeletevalue、
  - xp_regenumvalues、xp_regread、xp_regwrite、xp_loginconfig、xp_dirtree
- 检查扩展存储过程
  sp_helpextendedproc['procedure']
  例：sp_helpextendedproc'xp_cmdshell'
- 删除扩展存储过程
  sp_dropextendedproc['procedure']
  例：sp_dropextendedproc'xp_cmdshell'
- 恢复扩展存储过程
  sp_addextendedproc['procedure']['DLL']
  例：sp_addextendedproc'xp_cmdshell','C:\ProgramFiles\MicrosoftSQLServer\MSSQL\Binn\xplog70dll'

SQL的数据控制

- 完整性控制
  - SQL语言定义完整性约束条件的功能主要体现在CREATETABLE语句中，可以在该语句中定义约束条件。
- 并发控制
  - 当多个用户并发地对数据库进行操作时，采用封锁（lock）机制对其加以控制、协调，以保证操作正确执行和数据库的一致性。
- 数据恢复
  - 当数据库因故障而处于不一致状态时，具备恢复到一致状态的功能。SQL语言支持事务、提交（commit）、回滚（rollback）等概念。
- 安全性控制
  - 保护数据库，防止不合法的使用所造成的数据泄露和破坏。主要措施是对用户进行授权和访问控制。