Introduction of A Simplified Crypto Bank

冯镜维
李博文
高志成

2024 年 9 月 3 日

Menu

问题分解

银行转账

- 身份认证
  - 注册: 邮箱号,用户名,登录密码,支付密码
  - 登录: 用户通过输入邮箱和密码进行登录
  - 用户信息修改(忘记密码)
- 银行卡管理
  - 开通新银行卡
  - 多张银行卡，金额独立
- 交易
  - 存取款(模拟)
  - 转账

交易

方式

1. Alice与Bob直接通过网上银行进行转账
2. Bob开了一家牛肉烧麦店，挂在交易平台上，Alice下订单购买

简单商城

虚拟银行为虚拟商城提供必要的支付接口，虚拟商城用于模拟用户在实际生活中交易的过程。

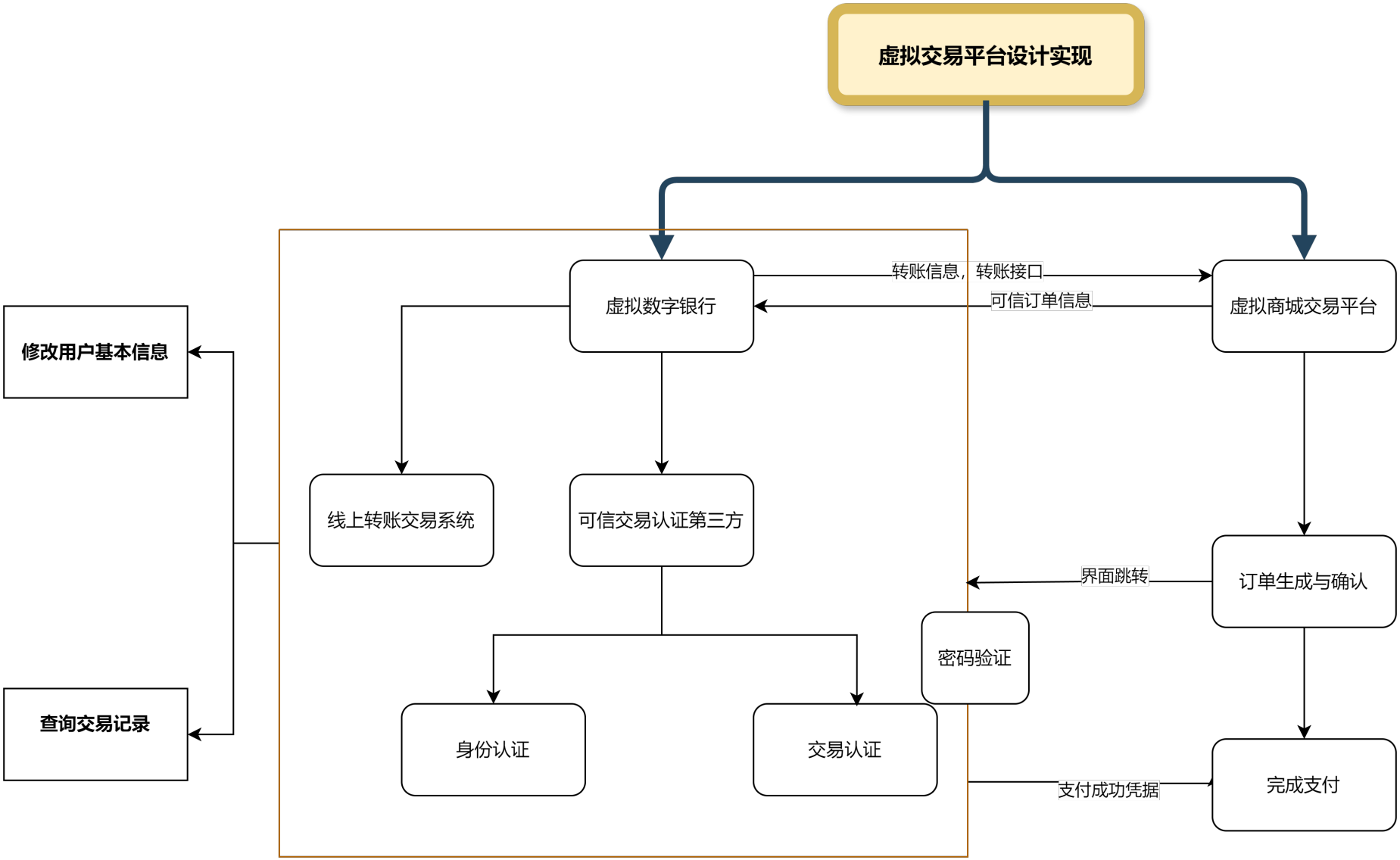
例如 Alice在Bob的网店上订购了一斤烧麦

- 商城：订单确实存在，而且交易双方的身份不能变
- 银行：可信资金平台，Alice确实有足够多的钱，这笔钱确实转给了Bob

模块总结

1. 利用银行来实现转账、存取款
2. 利用网络商城模块和银行模块的协作来实现交易
3. 银行和网络商城都需要一套安全的交互协议来保证可信

顶层结构图


虚拟交易平台设计实现
虚拟数字银行
转账信息，转账接口
可信订单信息
虚拟商城交易平台
修改用户基本信息
线上转账交易系统
可信交易认证第三方
界面跳转
订单生成与确认
密码验证
查询交易记录
身份认证
交易认证
支付成功凭据
完成支付

安全需求

从攻击者的角度看，哪些地方可以狠狠地攻击

- 用户信息认证 <- **unsign, cookie injection，弱口令爆破**
- 交易过程 <- **信息窃听, MIM, 篡改，重放**
- 数据处理
  - 数据查询过程 <- **SQL注入**
  - 数据渲染 <- **SSRF导致的RCE**

  此外，整个过程还涉及DDOS等一些暴力下三滥手段

安全需求

根据刚才列举的简单部分攻击，分析安全需求

1. 账户安全：采用**jwt**严格认证，规范过期时间，对session添加**唯一id**
2. 交易过程：对于商城交易，采用SET协议，对于银行转账，采用AES对交易信息加密，并SHA256生成摘要进行验证，**确保CCA安全**
3. 通信安全：使用TLS1.2防范窃听攻击和中间人攻击
4. 数据处理：采用前后端分离开发，在后端对前端发来的参数请求执行严格的白名单过滤策略
5. 用户信息安全：规范并强制用户提高密码复杂度以防范简单的弱口令爆破
6. 针对拒绝服务攻击：严格控制请求次数，构筑基本的防火墙

用户信息认证模块

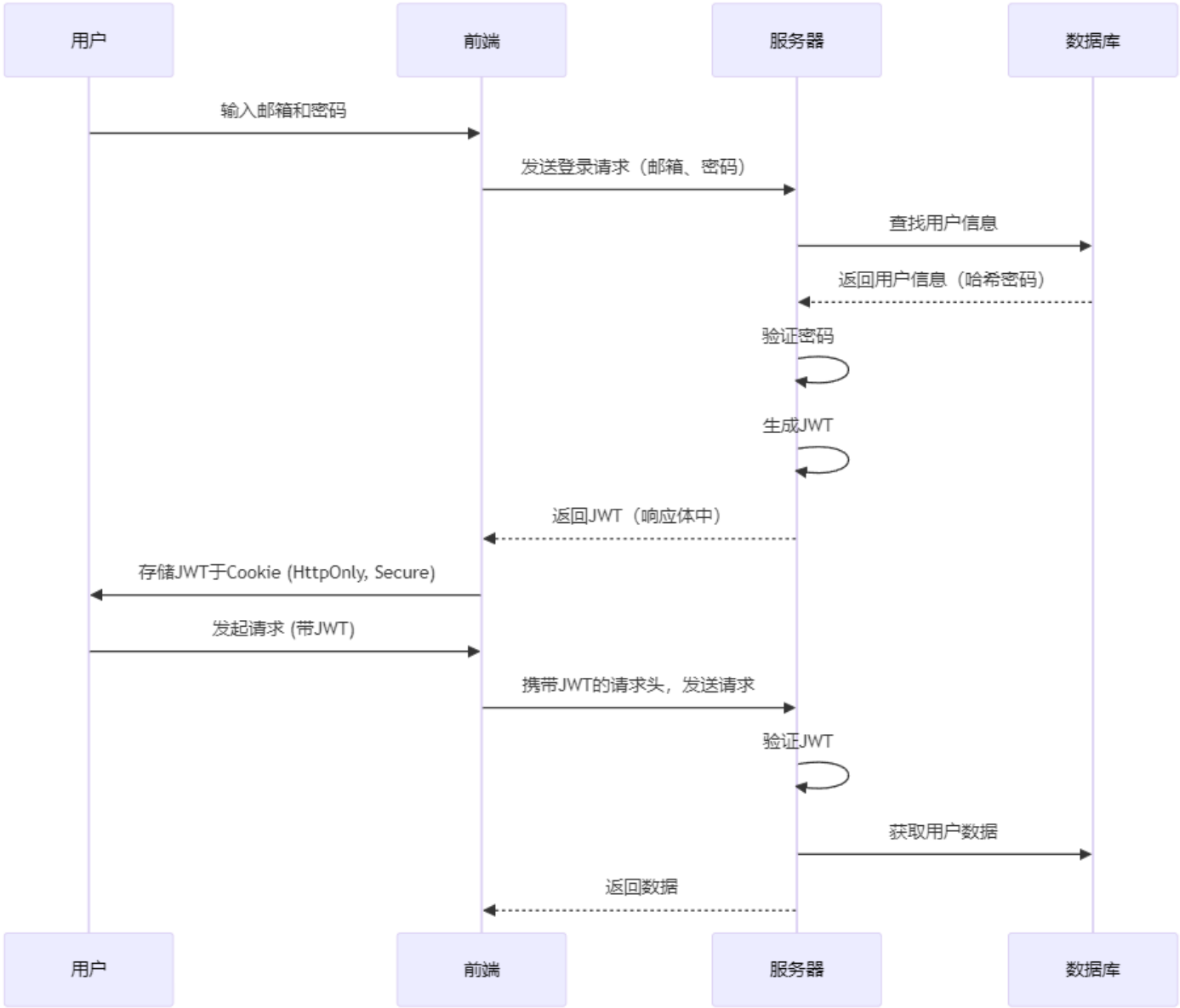
用户操作

- 注册：实名身份信息认证,登录密码,支付密码，输入唯一用户名以及邮箱，提供唯一用户ID
- 登录：用户通过唯一用户名登录，密码正确即可进入系统
- 忘记密码：通过邮箱重置账号密码

服务器认证过程

1. 对用户信息利用**jwt**进行封装
2. 对支付密码和登录密码用**SHA256**哈希
3. 前端设置好**cookie**并规范**ID**和**过期时间**

认证模块信息交互流程图



认证模块的额外要求

- 要有admin账户，对一切用户进行管理
- 要求用户有实名认证环节，调用其他可信接口（例如国家信息认证），在本项目中模拟为通过管理员审核
- 用户在实名认证前不能进行任何交易操作

网上交易模块

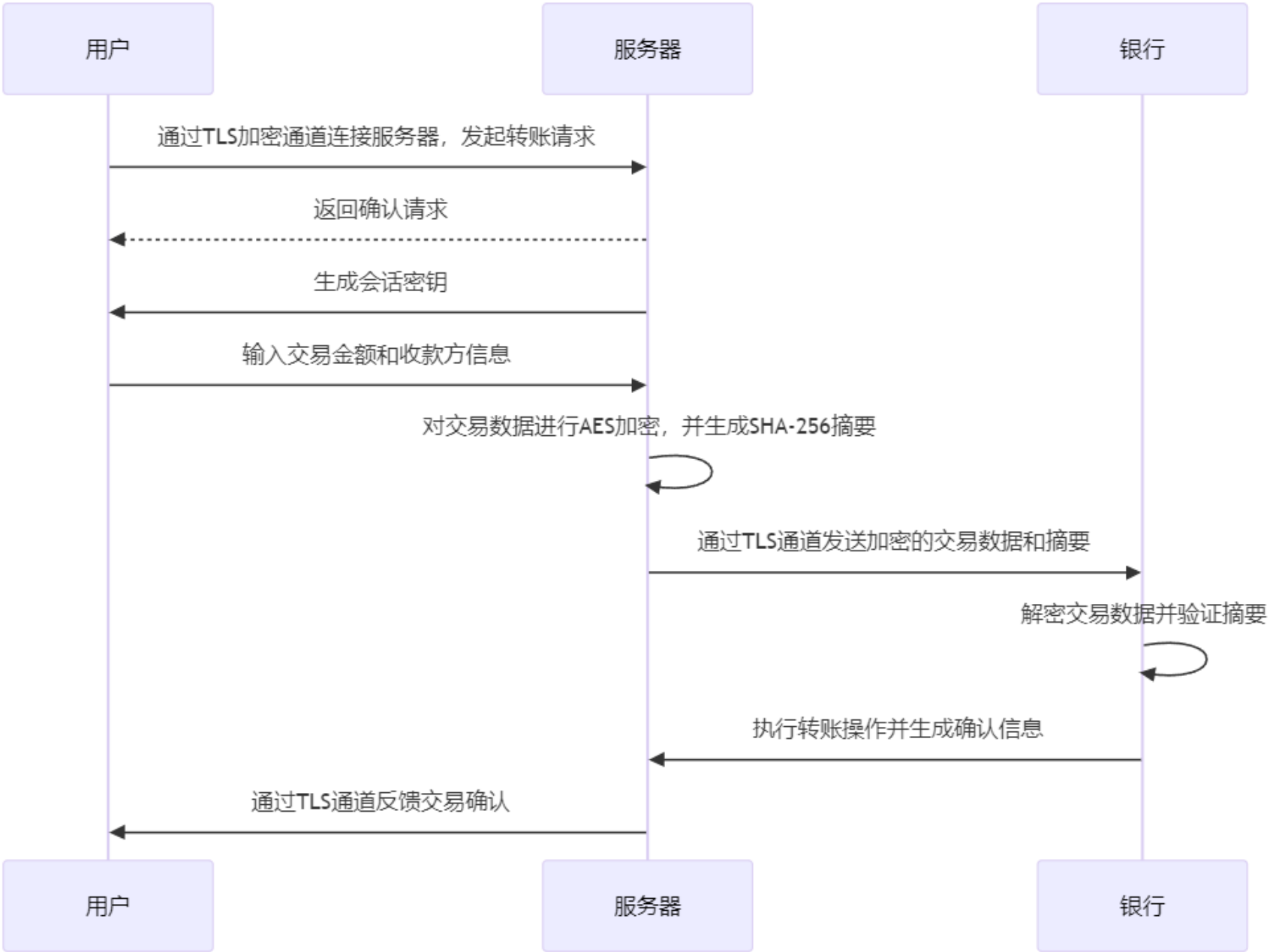
银行转账协议的安全设置

- **通信安全**：所有通信都通过**TLS协议**进行，确保数据在传输过程中的机密性、完整性和真实性，防止中间人攻击
- **交易加密**：每一笔交易都使用**AES加密**，确保交易数据在存储和传输过程中的安全性
- **消息摘要**：对交易数据进行**SHA-256摘要**处理，确保数据在传输和存储过程中未被篡改，防止重放攻击和选择密文攻击（CCA）

网上交易的安全流程

1. 用户通过TLS加密通道连接服务器，发起转账请求
2. 服务器生成会话密钥，用于本次交易的**AES加密**
3. 用户输入交易金额和收款方信息，服务器对数据进行**AES加密**，并生成**SHA-256摘要**
4. 将加密后的交易数据和摘要通过**TLS安全通道**发送给银行
5. 银行解密交易数据，验证摘要，确保数据的完整性和真实性，然后执行转账操作
6. 转账完成后，银行生成确认信息，并通过**TLS安全通道**反馈给用户，确认交易已成功处理

交易过程流程图


用户
服务器
银行
通过TLS加密通道连接服务器，发起转账请求
返回确认请求
生成会话密钥
输入交易金额和收款方信息
对交易数据进行AES加密，并生成SHA-256摘要
通过TLS通道发送加密的交易数据和摘要
解密交易数据并验证摘要
执行转账操作并生成确认信息
通过TLS通道反馈交易确认
用户
服务器
银行

存取款操作流程

1. **身份验证**：用户通过用户名和密码登录系统。
2. **银行卡选择**：用户选择要进行存取款操作的银行卡。
3. **银行卡验证**：服务器验证银行卡的真实性。
4. **余额检查**（仅取款）：系统检查所选银行卡的余额是否足够。
5. **存取款操作**：用户输入存款或取款金额，服务器处理操作并反馈交易确认。

存取款信息安全保障

- **通信安全**：使用TLS协议确保数据传输安全。
- **数据加密**：对敏感信息进行AES加密处理。
- **数据完整性**：使用SHA-256摘要验证数据完整性。
- **身份验证**：使用JWT严格验证用户身份。
- **会话安全**：为每次交易生成唯一的会话密钥。

额外约束

- 转账双方都需要有至少一张银行卡。
- 双方必须在实名认证后才可以申请绑定银行卡，因此在这之后才能转账。
- 转账方的银行卡余额必须充足。提款时用户对应的银行卡余额也必须充足。

虚拟线上商城

用户认证

线上商城只是为了模拟交易过程，因此为了简化用户部分，我们假定这个商城是银行系统的派生产品，因此共用一套用户系统。

同样，用户只有在实名认证之后，才有权利开张店铺并使用银行系统的各种功能和接口。

确保订单安全

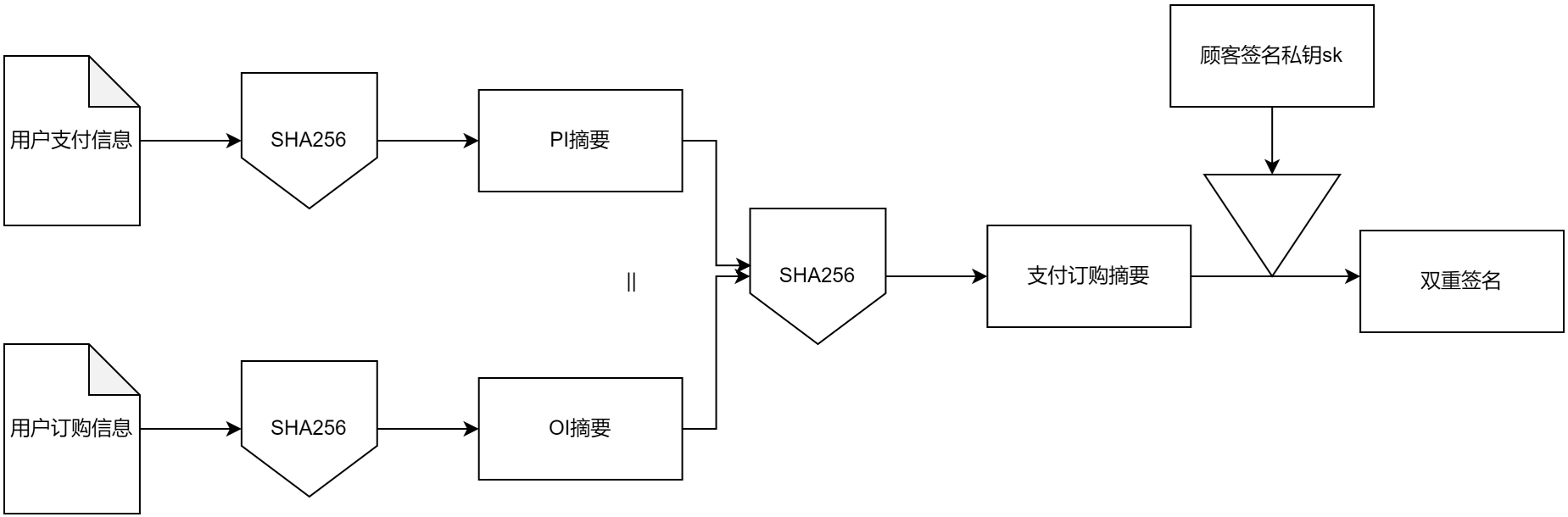
在订单交易过程中，银行在实质上是一个可信的第三方，用来证明付款人A的银行卡余额充足，且收款人B的银行卡确实存在，并能可信地通知交易平台转账成功，从而结束交易。

在这个过程中，银行模块**不能查看商家和客户之间的订单信息，只能查看转账信息**。同时要有**验证订单信息真实性的能力**。

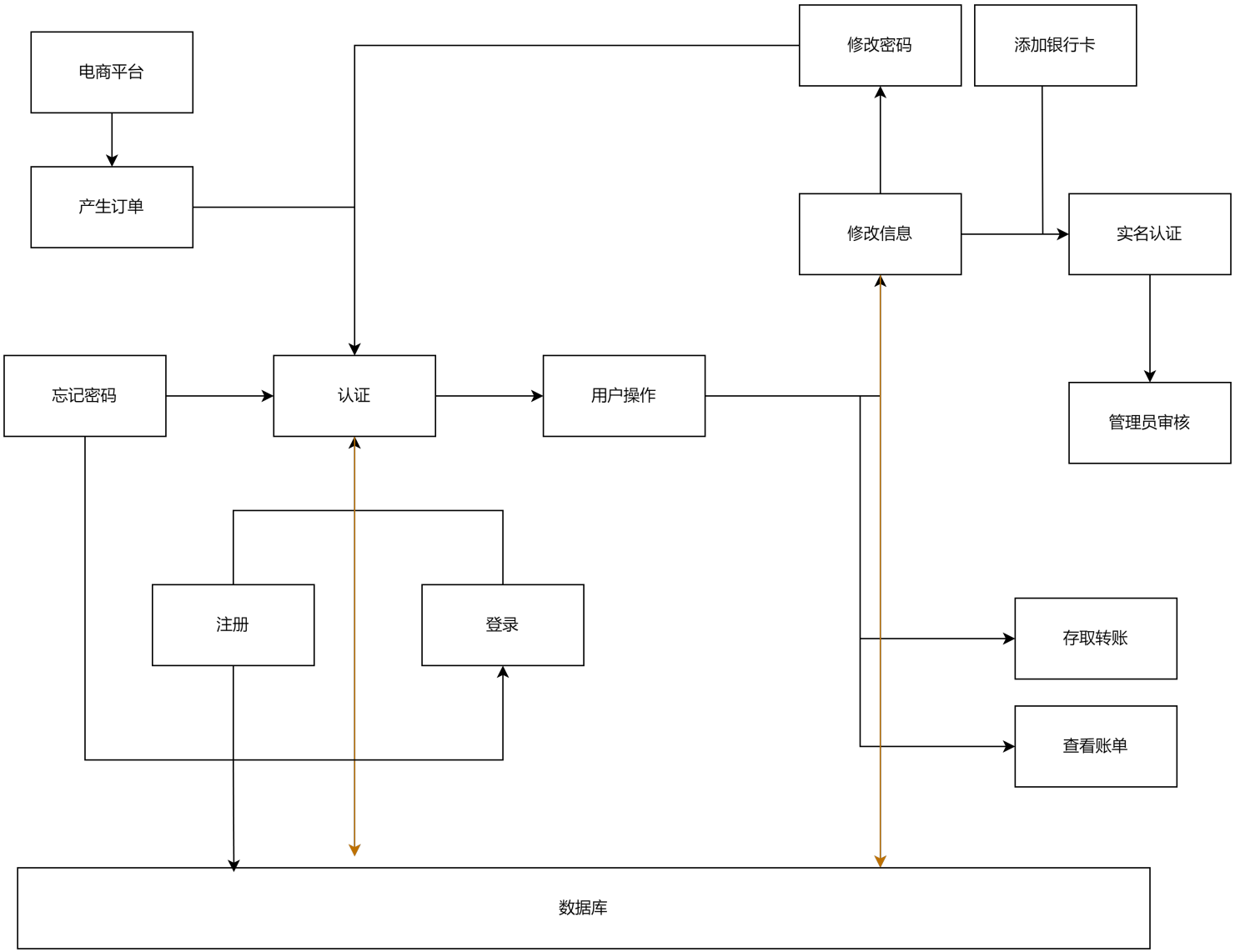
双签名与SET Protocol

前置要求

先审核订单，确保真实性，然后再执行支付操作，并将转账信息写入数据库，更新双方对应银行卡的余额信息。



简单业务数据流


电商平台
产生订单
修改密码
添加银行卡
修改信息
实名认证
忘记密码
认证
用户操作
管理员审核
注册
登录
存取转账
查看账单
数据库

技术栈

- 前端:vue+elementui+eleadmin
- 后端:flask (serve as an api) + SQLalchemy
- 数据库:mysql
- 加密库:pycryptodome