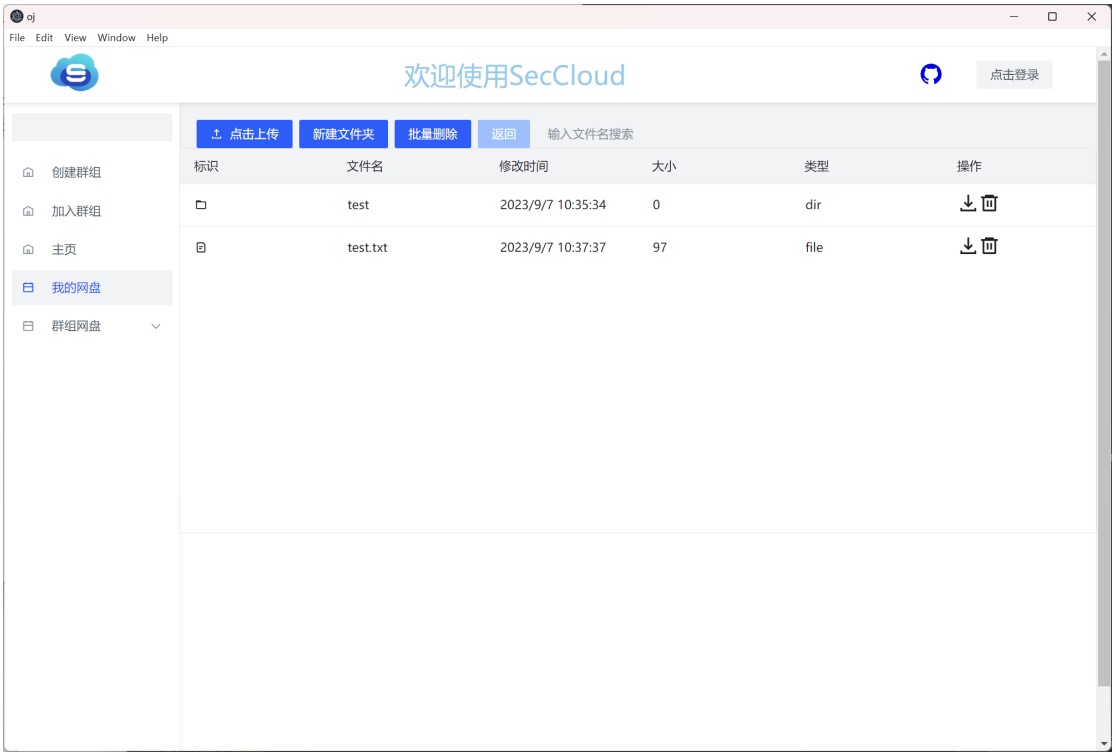
作品技术原理介绍

一、项目简介

SeCloud 是一款集安全认证、加密传输和加密存储于一体的高级网盘应用,专注于密码学设计,实现用户中心的加密、认证体制,SeCloud 服务器无法窃取任何用户网盘数据。用户可获得与传统网盘应用相近的网盘体验。该项目从底层构建安全网盘应用,自主设计实现客户端与服务器的通信协议,并在该通信协议上增加层层密码学协议及技术,实现了安全、可信、便捷的网盘应用。



二、系统架构设计

(一)客户端设计

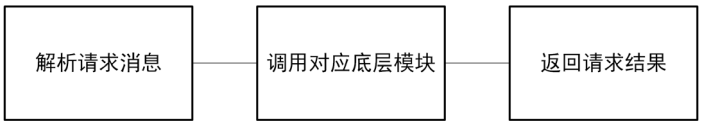
客户端采用前后端分离架构。客户端前端使用 Vue 框架,实现界面渲染、交互逻辑等功能。客户端后端使用 Python 语言,主要实现与服务器的通信、本地文件加密等功能。

客户端前端通过发起 HTTP 请求与本地后端通信（注意到。客户端后端与前端是在同一台主机的，不涉及网路通信），在收到请求后,会加工参数,通过自定义的安全传输协议与服务器通信。收到服务器响应后,后端对数据进行处理,然后将结果返回给前端。前端根据结果更新界面。

这种分离的设计使前后端职责清晰,也使得前端和后端可以独立扩展和调优。同时也更适合我们小学期团队的合作开发。

(二)服务器设计

1.客户端消息解析处理模块



该部分主要实现了对消息序列的解析、对底层模块的调用与请求结果处理的响应。

用户请求消息采用 json 编码传递至服务器。服务器通过解析其中字段获取具体的指令与数据，并进行相应操作。每条请求均有"op"字段，用于标识请求的操作名。不同的指令有不同的附加字段，用于提供请求所需的数据。具体如下表所示。

操作名	说明	附加字段
gen-authcode	发送验证码	email
pwd-login	密码登录	email, pwd
authcode-login	验证码	email, authcode
update-pwd	重置密码	email, pwd, authcode
register	注册	email, pwd, authcode
get-dir-list	获取目录列表	id, path
put-file	上传文件	id, path
create-dir	创建目录	id, path
del-dir	删除目录	id, path
get-file	下载文件	id, path
join-group	加入群组	id

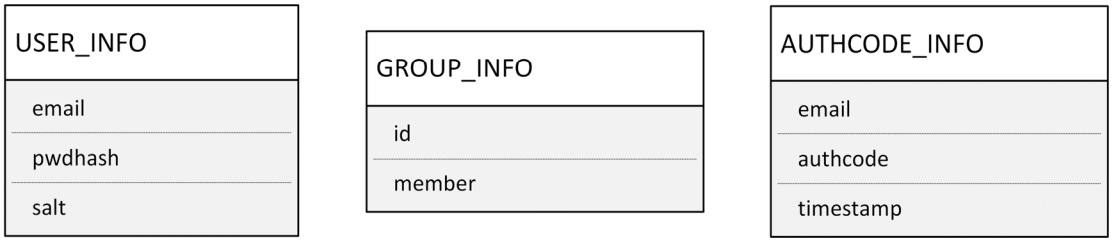
在调用底层模块获得结果后，服务器再将相应消息编码成 json 字符串。每个响应中都会附上"status"字段，请求正常为 200，请求异常为 400。

2.文件存储与检索模块

对于每一个个人空间和群组空间，都在存储路径下有一个专用文件夹。其中，个人空间以 16 进制的用户邮箱 MD5 值作为目录名，群组空间以 16 进制的群组标识符作为目录名。每个空间的文件都存储在相应的目录下。

每个空间下的目录结构和文件元数据（如修改时间、文件大小等）都由操作系统的文件系统提供。模块通过直接调用操作系统的接口获取相关信息，并按规定格式编码后传输回客户端进行展示。

3.用户信息数据库模块



图：数据库模式

用户信息数据库模块用于存储并维护用户及群组相关的各种信息。数据库由三张表构成。其中，用户信息表（USER_INFO）用于存储用户的邮箱（email）、口令哈希值（pwdhash）与口令盐值（salt）信息；群组信息表（GROUP_INFO）用于存储群组标识符（id）、群组成员邮箱（member）信息；验证码信息表（AUTHCODE_INFO）用于存储指定邮箱（email）的验证码（authcode）与发送时间戳（timestamp）信息。

(三)　客户端与服务器通信设计

1.C/S 端到端透明加解密传输协议

客户端与服务器之间采用自定制的安全传输协议。该协议工作在 TCP 传输层之上,以提供加密、认证、完整性保护等功能。

工作流程:

a. 通过 CA 验证建立 TLS 通道,保证传输加密;
b. 利用 DH 算法协商一条对话会话密钥;

c. 使用会话密钥加密应用数据,以 AES-256 算法发送;

d. 每条应用数据附带初始化向量,确保不可重复;

e. 数据包中包含 HMAC 码,验证内容完整性。

f. 这种安全传输协议设计,保证了客户端与服务器之间数据的保密性、完整性和端到端加密传输。

通过这种 socket 编程的 TCP 传输，我们实现了客户端与服务器端到端的透明加解密。在此技术上，我们借鉴 HTTPS 的 C/S 与安全设计，基于 TCP 设计了 C/S 端到端透明加密通信协议。客户端与服务器的每次通信都是建立会话密钥的加密通信。

(四) 文件加密设计

我们的安全网盘系统存在三类“密码”：**用户口令、会话密钥、文件加密密钥**。前文介绍客户端与服务器通信设计时，已经介绍了会话密钥部分，现在介绍文件加密密钥。

我们实施了一种基于硬件安全模块（U 盾）的文件主密钥管理机制，并采用了层级密钥生成策略，保证每个文件的加密密钥都不相同。

设计层面	描述
设计原因	1. 为确保数据的安全性和完整性，避免数据泄露和篡改 2. 强化密码安全，保障密码的强度和随机性的同时方便密钥管理 3. 既要单用户多终端登录实现文件解密密钥同步，又要阻止服务器解密文件而不能存储密钥在服务器上
设计方法	1. 利用PBKDF2算法结合盐值生成随机主密钥 2. 使用AES在CTR模式进行文件加密，CFB模式构建安全信道 3. 通过HMAC确保文件的完整性和安全性 4. 主密钥物理存储在U盘中，与网络隔离以增强安全性和便携性 5. 主密钥命名采用随机数或者MD5(email)命名，增加其安全性，减少主密钥信息泄露
安全性证明	1. PBKDF2算法可以防止彩虹表攻击，增强密码的安全性 2. AES加密算法提供了高级的数据加密保障，确保数据的保密性 3. HMAC验证可以防止数据被篡改，保证数据的完整性 4. U盘物理存储主密钥能有效阻止网络攻击，提高系统的安全性
防御攻击	1. 防御中间人攻击和重放攻击，通过构建安全信道和验证文件完整性实现 2. 防止密钥猜测攻击，通过PBKDF2和强随机数生成方法减少这种可能性 3. 防止网络攻击，通过U盘物理存储主密钥实现网络和物理隔离。

通过以上基于硬件安全模块（U 盾）的文件主密钥管理机制，我们实现了文件的安全，服务器、攻击者除非拿到了 U 盾以及用户口令，否则不可能窃取用户网盘资料。

三、密码学核心技术介绍

(一)CA 数字证书模块

系统中模拟了一个 CA 中心,用于签发数字证书。服务器端拥有 CA 签名的证书,客户端可以验证服务器证书的有效性,确保通信过程中服务器的身份。数字签名技术也用于保护数据完整性。以下是证书各步骤的设计原因、方法、安全性分析：

1.CSR 文件设计

CSR 文件用于请求证书，以下是设计分析：

设计层面	描述
设计原因	用于证明请求者的身份，确保其是合法的证书请求者 为动态生成数字证书提供一种机制，使得每一个实体都能有其独特的证书
设计方法	请求者生成一对公私钥 生成CSR文件，包含公钥和一些身份信息
安全性证明	私钥只在请求者处 SHA-256算法用于确保信息的完整性和未被篡改
防御攻击	防止身份冒充攻击 防止信息篡改

2.数字证书设计

数字证书用于保证服务器的真实性，防止被身份冒充、中间人攻击。

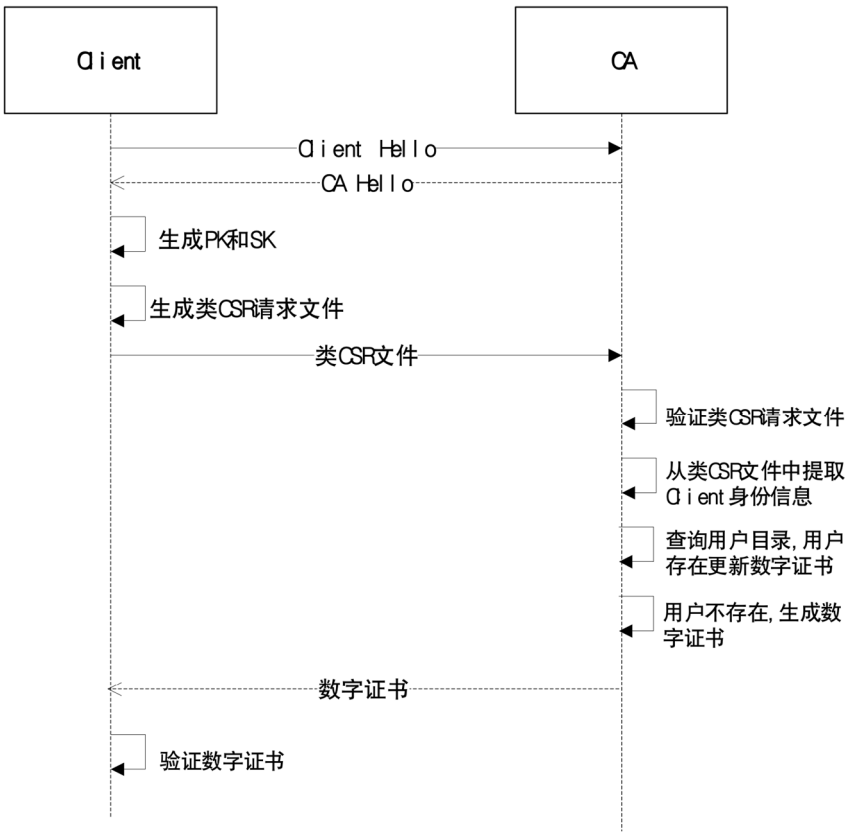
设计层面	描述
设计原因	为实现安全通信提供身份验证手段 建立可信任的身份验证机制
设计方法	CA机构对CSR文件进行审核 CA使用其私钥签名生成数字证书
安全性证明	证书的签名可由CA的公钥验证，保证了证书的真实性和合法性
防御攻击	防止中间人攻击 防止身份冒充

3.验证数字证书方法设计

客户端拿到数字证书后，需要对数字证书进行验证。

设计层面	描述
设计原因	确保通信双方的身份真实性 验证数字证书的有效性和合法性
设计方法	通过CA的公钥验证数字证书的签名 验证证书的有效期
安全性证明	能有效识别和拒绝伪造或篡改的证书
防御攻击	防止中间人攻击 防止证书伪造和篡改

4.数字证书操作流程图：



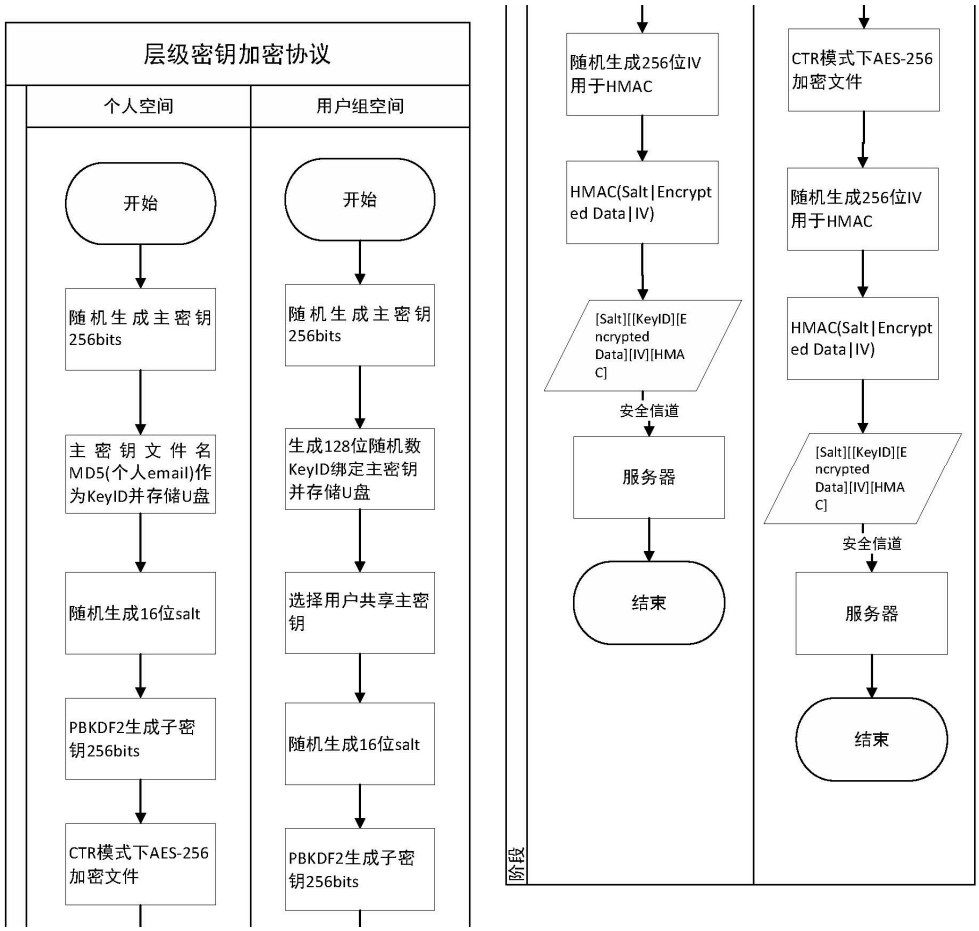
(二)层级密钥体系（文件加密）

系统采用主密钥-会话密钥-文件密钥的三层密钥体系。主密钥由硬件安全模块管理,用于导出会话密钥;会话密钥用于客户端和服务器间通信加密;文件密钥对每个文件独立生成,实现了一次一密。

我们使用 U 盾的文件加密密钥（主密钥），对每个文件生成不同的文件加密密钥，我们的层级密钥设计思路如下：

设计层面	描述
设计原因	1. 为确保数据的安全性和完整性，避免数据泄露和篡改 2. 强化密码安全，保障密码的强度和随机性的同时方便密钥管理 3. 既要单用户多终端登录实现文件解密密钥同步，又要阻止服务器解密文件而不能存储密钥在服务器上
设计方法	1. 利用PBKDF2算法结合盐值生成随机主密钥 2. 使用AES在CTR模式进行文件加密，CFB模式构建安全信道 3. 通过HMAC确保文件的完整性和安全性 4. 主密钥物理存储在U盘中，与网络隔离以增强安全性和便携性 5. 主密钥命名采用随机数或者MD5(email)命名，增加其安全性，减少主密钥信息泄露
安全性证明	1. PBKDF2算法可以防止彩虹表攻击，增强密码的安全性 2. AES加密算法提供了高级的数据加密保障，确保数据的保密性 3. HMAC验证可以防止数据被篡改，保证数据的完整性 4. U盘物理存储主密钥能有效阻止网络攻击，提高系统的安全性
防御攻击	1. 防御中间人攻击和重放攻击，通过构建安全信道和验证文件完整性实现 2. 防止密钥猜测攻击，通过PBKDF2和强随机数生成方法减少这种可能性 3. 防止网络攻击，通过U盘物理存储主密钥实现网络和物理隔离。

流程图如下：（右图接左图）

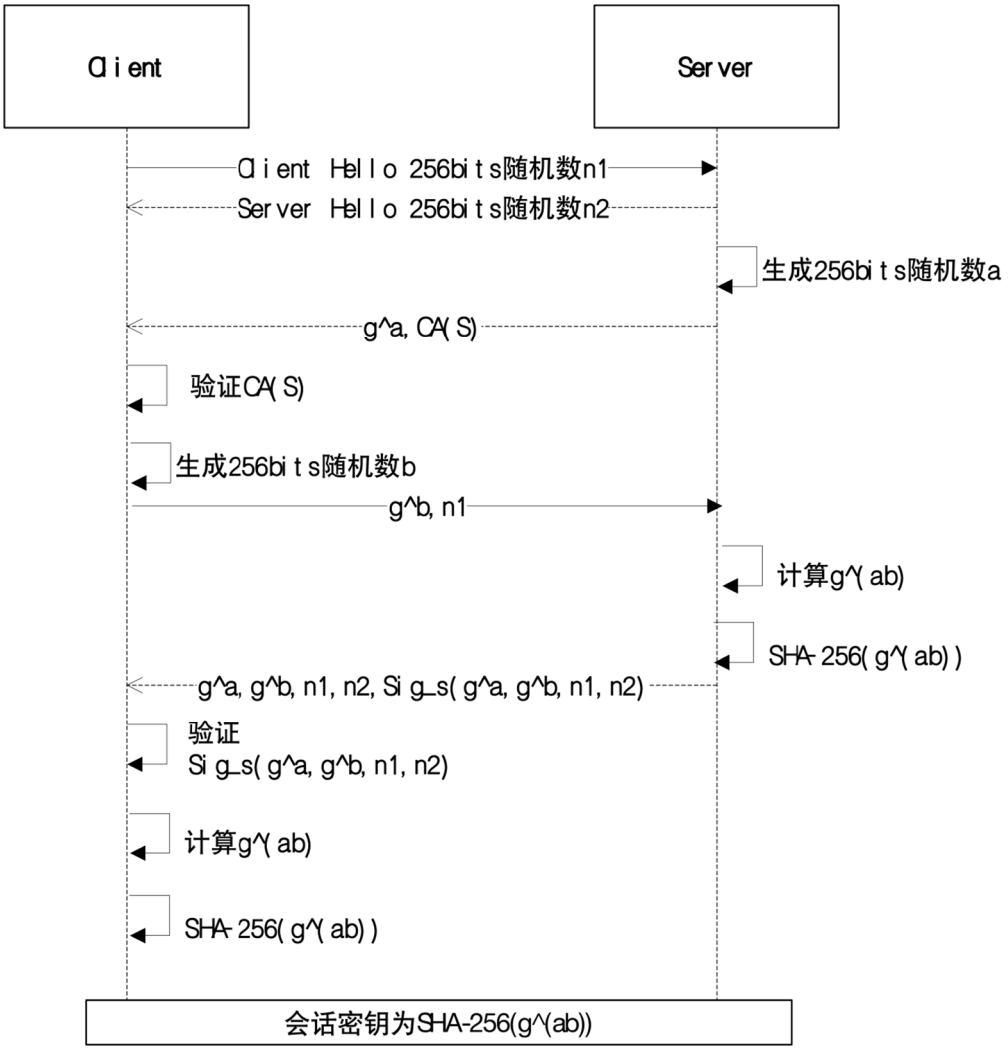


(三)DH 密钥协商（协商会话密钥）

客户端和服务器之间通过 DH 算法协商保护通信的对称会话密钥。该算法可以防止中间人攻击,保证密钥的安全性。协商后的会话密钥用于后续通信的加密。

设计层面	描述
设计原因	1.为实现安全的通信会话，确保信息在传输过程中的保密性和完整性 2.利用DH算法中的离散对数难题，建立一种安全的密钥交换机制
设计方法	1.利用随机数避免重放攻击，通过每次会话生成新的随机数来确保安全性 2.服务器提供数字证书进行身份验证 3.应用DH算法进行密钥交换，利用离散对数难题为基础来保证交换过程的安全 4.利用私钥签名验证会话参数的正确性和安全性 5.使用SHA-256对g^(ab)进行哈希，以缩短其大小并保证传输的效率和安全性
安全性证明	1.DH算法中的离散对数难题可以保证即使攻击者知道公共参数和公钥，也很难计算出私钥 2.数字证书和私钥签名可以验证服务器的身份，确保不是与冒充者通信 3.SHA-256可以有效防止碰撞攻击
防御攻击	1.防止重放攻击，通过使用会话随机数来保证每次会话的唯一性 2.通过数字证书和私钥签名验证来防止中间人攻击和身份冒充

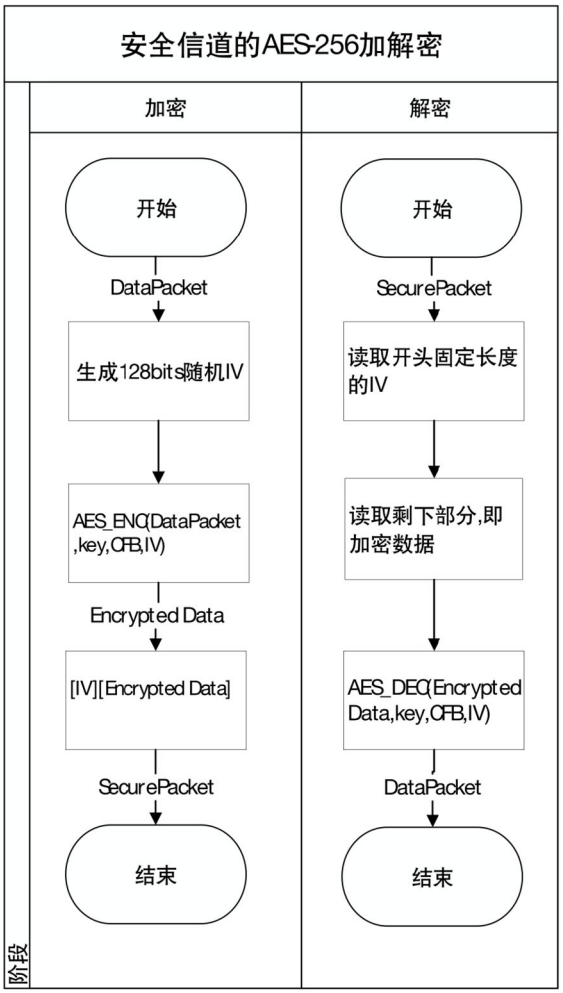
DH 密钥协商流程图如下：



(四)安全通信协议（会话密钥加密通信）

设计层面	描述
设计原因	1. 为确保信道内容的安全性和完整性，防止数据在传输过程中被未授权访问和篡改 2. 自主设计的文件格式可以便于快速和准确地提取关键信息来初始化AES，从而优化加解密过程的效率
设计方法	1. 使用AES算法对信道内容进行加密，确保数据安全 2. 利用强随机数生成初始化向量(IV)，增强加密的安全性 3. 自主设计的文件格式能够方便地提取关键信息，用于AES的初始化。
安全性证明	1. AES加密算法已被公认为具有很高的安全性，能有效防止多种攻击 2. 强随机数IV可以保证每次加密的唯一性，增强系统的安全性
防御攻击	1. 可以防御中间人攻击，保护数据不被篡改或窃取

自定义的传输协议运行在 TCP 之上,所有数据使用协商的会话密钥加密,每个数据包有一个随机的 IV。加密采用 AES 算法,检验采用 HMAC,使数据传输具有保密性和完整性。



上图为流程图，使用会话密钥进行安全数据包通信流程如下：

加密部分：

Step 1: 加密初始化

1.接收 DataPacket 和会话密钥，准备进行 AES-256 加密处理

2. 生成一个随机的初始向量（IV）

Step 2: 文件加密

1.使用协商好的会话密钥和生成的随机 IV，初始化 AES-256 加密器在 CFB 模式下

2.将 DataPacket 进行 AES-256 CFB 模式加密，生成加密后的数据块 Encrypted Data

Step 3: 构建安全数据包

1.将生成的随机 IV 放在新的数据包 SecurePacket 的起始位置

2.接着将加密后的数据块附加到 IV 后面，完成 SecurePacket 的构建

3.最终 SecurePacket 的格式将如下所示：[IV][Encrypted Data]

解密部分

Step 1: 解密初始化

1.接收 SecurePacket，这是一个经过 AES-256 CFB 模式加密的数据包

2.分别从 SecurePacket 中提取出初始向量（IV）和加密数据块（Encrypted Data）

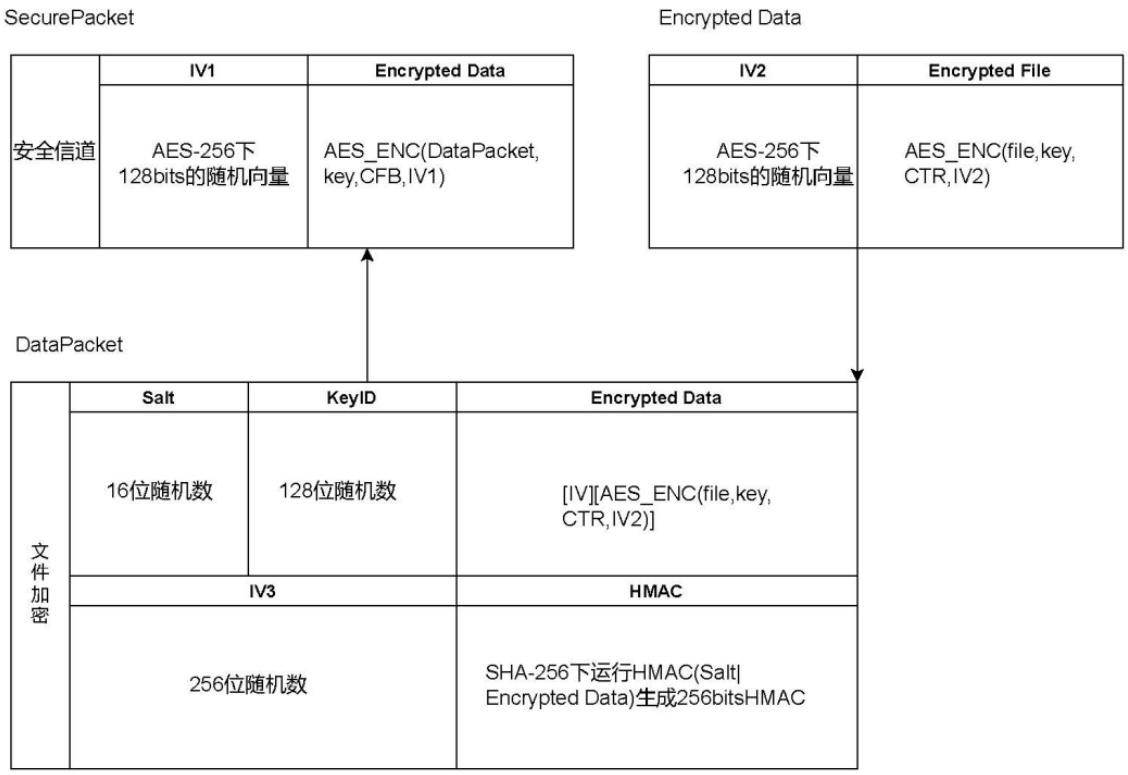
Step 2: 会话密钥与初始化向量准备

1.使用之前协商好的会话密钥，结合提取出的 IV，为 AES-256 解密器在 CFB 模式下进行初始化

Step 3: 文件解密

1. 使用初始化完成的 AES-256 CFB 模式解密器，对提取出的 Encrypted Data 进行解密得到解密后的原始文件内容，即 DataPacket

（五）文件结构总览



数据首先被文件加密，然后被信道层加密，上图即为相应阶段文件结构。

八、项目亮点

- ✓ **用户中心的加密和认证体制：**SeCloud 服务器无法窃取任何用户数据，从而保证了用户的隐私和数据安全。
- ✓ **基于 U 盾的多重安全机制：**通过硬件安全模块（U 盾）和多种密钥管理机制，增加了系统安全性。
- ✓ **一次一密的文件加密管理：**采用层级密钥结构和 U 盾生成的主密钥，确保每个文件都有独一无二的加密密钥。
- ✓ **端到端的透明加密传输：**利用 DH 密钥交换协议和数字证书，实现了与 HTTPS 相似的加密通信信道。
- ✓ **P2P 加密信道和 U 盾物理传递的共享网盘设计：**确保即使在多用户共享环境中，每个文件的加密密钥也都得到了安全传输和存储。
- ✓ **双重保护的登录认证机制：**除了传统的口令登录，还需要 U 盾进行文件下载，大大增

加了安全性。

- ✓ **完整性验证和中间人攻击防护：**通过使用 HMAC 和可信的第三方 CA，进一步保证了数据和通信的安全性。
- ✓ **灵活的加密数据找回机制：**通过与安全机构（如银行）合作，用户可以在 U 盾丢失后经过全方位验证后拿到备份，保证数据不会丢失。
- ✓ **适应多端登录的设计：**借鉴银行系统，将文件加密密钥存储在 U 盾中，实现多端登录时的高级安全同步。