门限加密和签名threshold\_crypto实操

* 前置条件

在本地安装rust环境，因为threshold\_crypto是用rust语言开发的，具体怎么安装rust环境就略过了。

* 理论基础

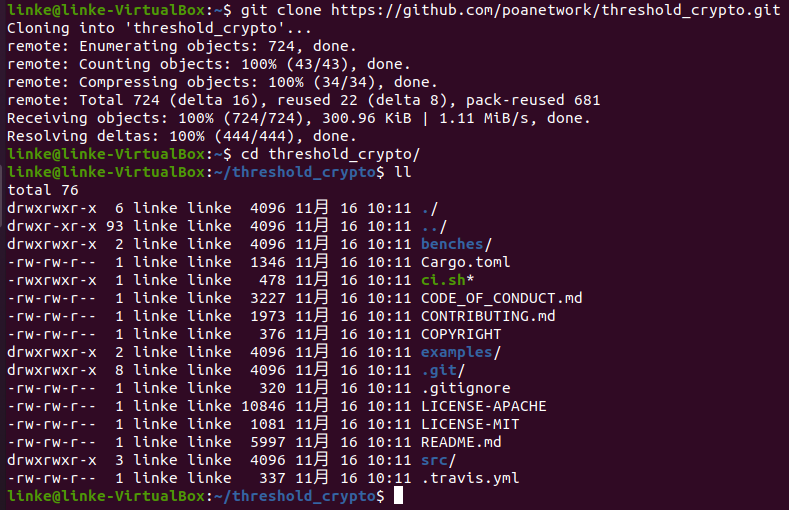
可搜索一些关于门限密码学的知识自学，推荐一篇文章

<https://mp.weixin.qq.com/s?__biz=MzU0MDY4MDMzOA==&mid=2247484866&idx=1&sn=5e1b428d532ac971beb8570dd15c648f&scene=19#wechat_redirect>

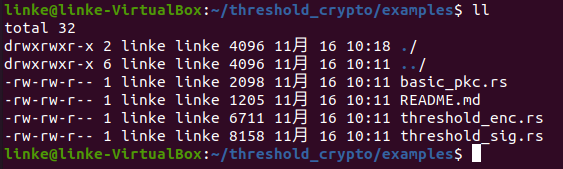
* 实操过程

首先把threshold\_crypto开源框架克隆到本地

git clone <https://github.com/poanetwork/threshold_crypto.git>



我们首先把examples目录下的示例程序都跑一遍



可以看到，有三个.rs文件，即三个示例程序。在跑示例程序之前，先对每个示例程序进行一个解读。

* 基本的加解密和签名验签

basic\_pkc.rs: 演示如何生成随机私钥和相应的公钥，使用私钥对字节进行签名，使用公钥验证字节的签名，使用公钥加密字节，以及使用私钥解密密文。

basic\_pkc.rs的代码我全部加了中文注释，如下



接下来运行一下basic\_pkc.rs

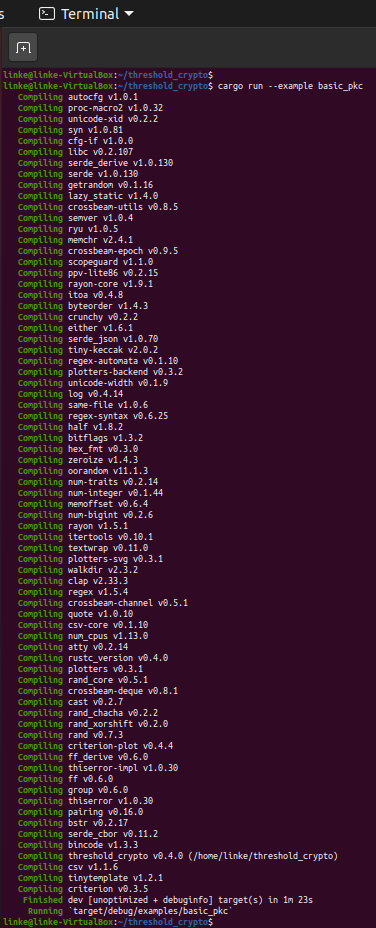
执行

MLOCK\_SECRETS=false cargo run --example basic\_pkc

或者

cargo run --example basic\_pkc

加上MLOCK\_SECRETS=false是显式地禁止密钥mock



没有报错，说明basic\_pkc程序中的断言都是正确的，没有错误，程序运行成功。

* 主公钥加密，参与者联合解密

接下来解读下一个程序threshold\_enc.rs

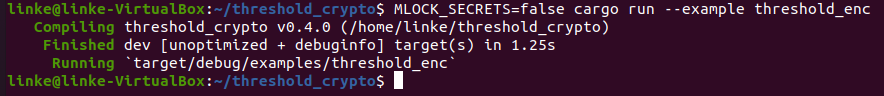
演示了如何使用master公钥将消息加密后发送给一组参与者，其中参与消息解密过程的参与者数量必须超过给定的阈值，才能成功解密消息密文。这个示例还演示了“trust dealer”的概念，即负责生成密钥的可信实体。

同样，我通读了源代码，并做了详细的中文注释，如下



然后运行这个示例程序，执行

MLOCK\_SECRETS=false cargo run --example threshold\_enc



没有异常，即所有的断言都正确，运行成功。

* 门限签名：多方联合签名，主公钥验签

演示如何使用门限签名生成聊天消息的可追加账本。运行聊天协议的每个节点接收消息并对消息进行签名(使用节点的私钥share)。一旦有足够多的节点(阈值+ 1)签名了给定的消息，网络就会向账本中添加一条新消息。

总体的逻辑是：

1. 初始化一个聊天网络，有3个节点。给定一个签名阈值（比如1，就代表比如1+1方签名才能够正确验签），通过随机数方式生成私钥集合和公钥集合（即主公钥），并给每个节点分配对应的私钥份额和公钥份额。
2. 用户发送一条消息给一个节点，节点对消息进行签名，然后用主公钥验签，发现验签不通过，因为签名的节点还没有达到阈值。
3. 用户把这条消息又发送给另一个节点，节点对消息进行签名，然后用主公钥验签，验签通过，因为签名的节点已经有2个，达到了阈值。

具体的代码细节和密钥生成、签名、验签方法的使用，请参考我对threshold\_sig.rs的详细中文注释，主流程在main方法中。在项目中如果有门限签名的需求，可以直接使用

图形用户界面, 文本, 应用程序

描述已自动生成

然后运行这个示例程序，执行

MLOCK\_SECRETS=false cargo run --example threshold\_sig

文本

描述已自动生成

电脑萤幕画面

描述已自动生成

没有异常，说明fn main()中的断言全部成功。

个人认为threshold\_crypto这个门限签名库非常有用。