

区块链 Project 最终报告

陈铭涛

16340024

1. 源码 GitHub 地址

智能合约：<https://github.com/miguch/cryptochat>

Web 前端：<https://github.com/miguch/cryptochat-web>

2. 选题背景、依据

DApp 名称：CryptoChat

背景与依据：

当前大部分的聊天应用如微信、WhatsApp 等都需要通过中心化的服务器来实现各个用户之间的通信，这带来了聊天信息可能被监视、篡改的风险。即使是 Telegram 这类以安全的加密通信为主要特点的聊天应用，也无法避免发起 P2P 通信时需要从中心化的服务器获取聊天对象的基本信息，从而可能导致 IP 地址等信息泄露的风险。本应用希望利用区块链技术，将聊天数据保存至区块链上，使聊天应用实现去中心化。利用 RSA 非对称加密算法，可以使保存在区块链上的聊天信息数据的明文信息只能被持有私钥的用户所看到，区块链的特征也使得数据极难以被篡改或删除，实现一定的通讯信息安全性。

3. 应用设计

应用使用 Solidity 编写智能合约，主要功能是对用户的用户名、公钥信息、信息签名以及发送与接收的消息密文进行存储。

应用前端界面以 Web 形式呈现，可在有 web3 注入的浏览器中运行，已在安装有 Metamask 的 Chrome 和 Firefox 浏览器以及 iOS 的 Cipher 应用中进行过测试。

前端界面部分使用 Vue 作为框架，UI 组件库使用 BootstrapVue。对消息的加密使用 Go 语言的 crypto 库，并利用 gopherjs 编译为 js 进行调用。

应用使用的加密算法为 AES 与 RSA，当访问用户的地址不存在对应用户时提示注册，根据注册的用户名与密码利用 PBKDF2 为随机函数用于生成 1024 位 RSA 密钥对，并将公钥存入区块链。当访问用户为已注册用户时，提示输入密码，根据输入的密码重新使用上述的方法生成密钥对，与区块链上的公钥进行签名验证，若通过则登录成功。

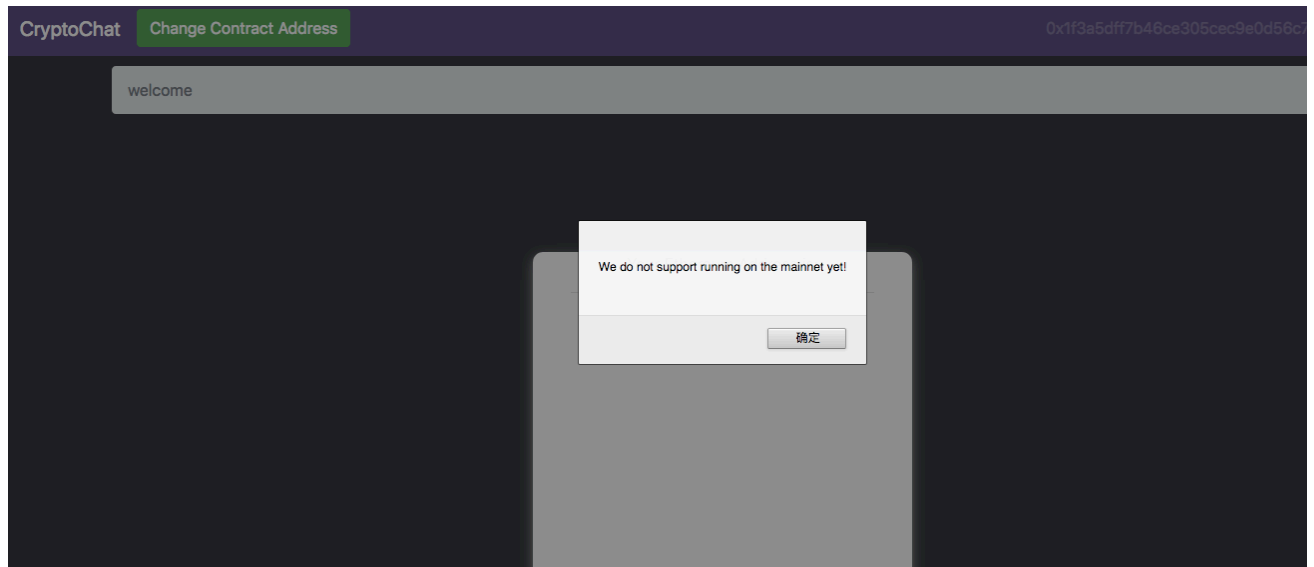
当用户发送信息时，将随机生成 32 位 AES 密钥，使用该密钥对消息进行加密后将密钥使用接受者的公钥进行 RSA 加密，将密钥的密文置于消息密文的头部后存入区块链。当用户从区块链获取接收或发出的消息时，将密文的前 128 位取出使用私钥解密获得 AES 密钥，然后对剩余密文使用 AES 解密获得消息的明文数据。

4. 应用使用方法

应用合约目前已部署至 Ropsten 和 Rinkeby 测试网络，可以直接通过

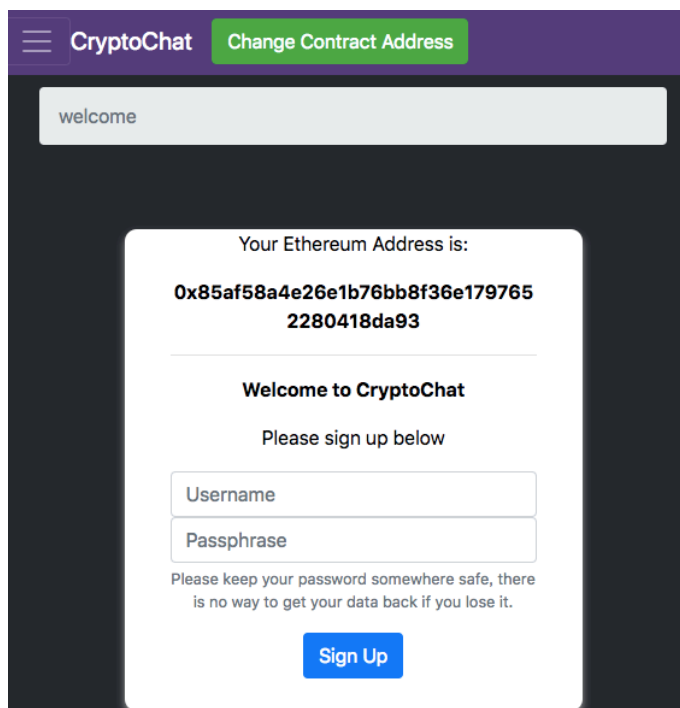
<https://cryptochat.miguch.com/> 访问。

在应用启动时，根据 network id 识别网络，若识别当前网络为主网或 Koven 测试网络，会通过 alert 报错：



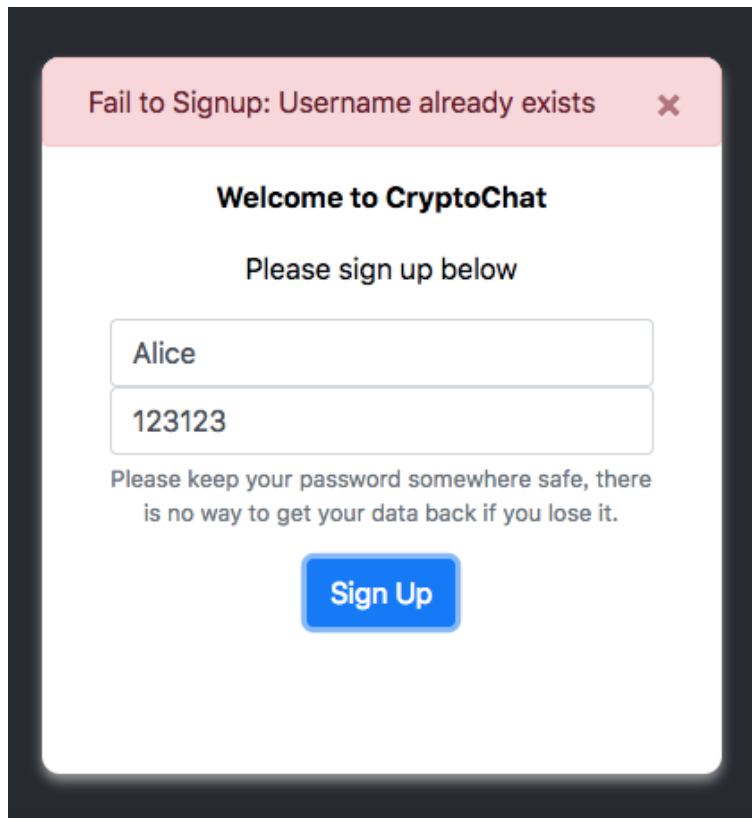
若为未识别的网络，则会使用开发时私链上的合约地址作为合约调用时使用的地址。

若合约地址可调用，则会根据当前地址是否存在用户显示注册或登录界面：



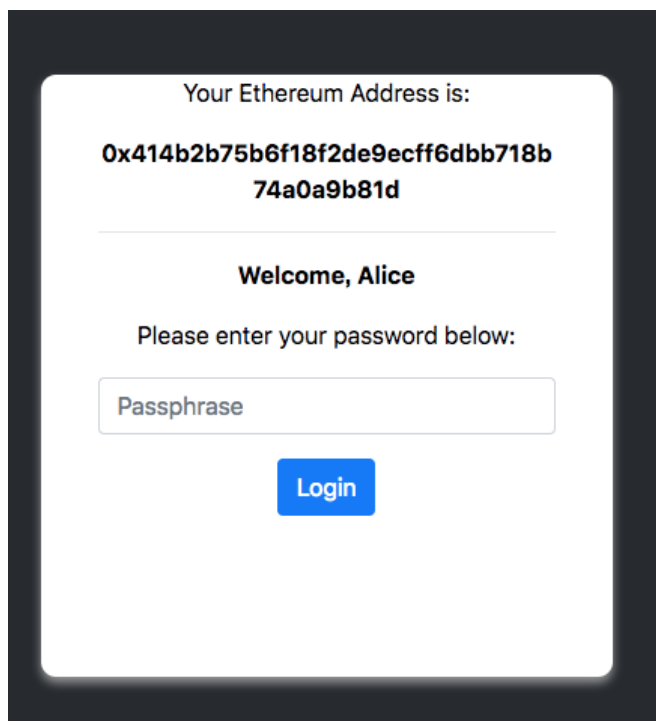
注册时需要输入用户名和密码，用户名不可与当前已存在的用户重复，否则提示

错误信息：



The screenshot shows a dark-themed interface with a white sign-up form. At the top, a pink error banner reads "Fail to Signup: Username already exists" with a close icon. Below the banner, the text "Welcome to CryptoChat" is centered, followed by "Please sign up below". The form contains two input fields: the first contains "Alice" and the second contains "123123". Below the fields, a warning message states: "Please keep your password somewhere safe, there is no way to get your data back if you lose it." At the bottom of the form is a blue "Sign Up" button.

若用户为已注册用户，进入应用时显示登录界面，要求输入密码：

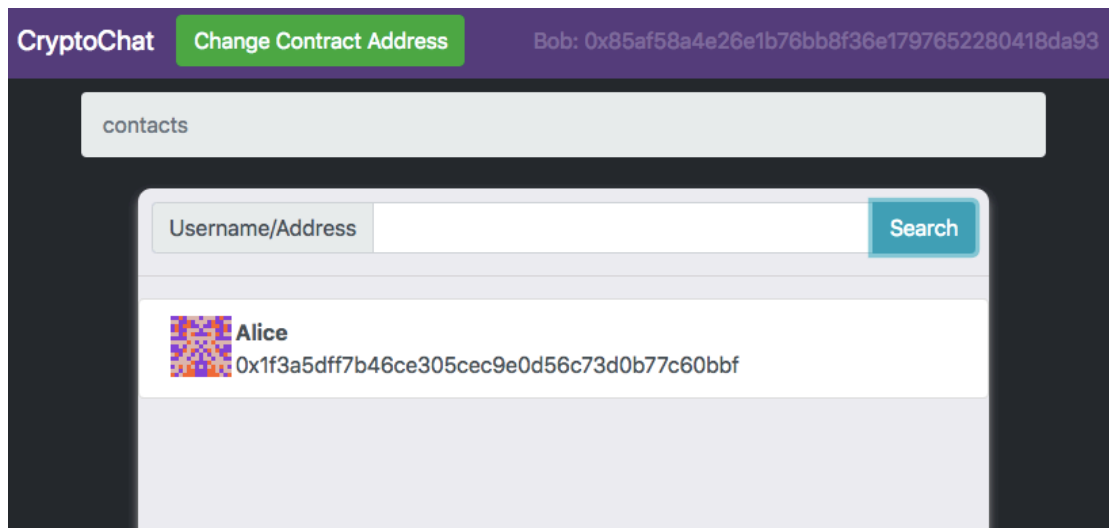


The screenshot shows a dark-themed interface with a white login form. At the top, it says "Your Ethereum Address is:" followed by the address "0x414b2b75b6f18f2de9ecff6dbb718b74a0a9b81d". Below this is a horizontal line. The text "Welcome, Alice" is centered, followed by "Please enter your password below:". The form contains a single input field labeled "Passphrase". At the bottom of the form is a blue "Login" button.

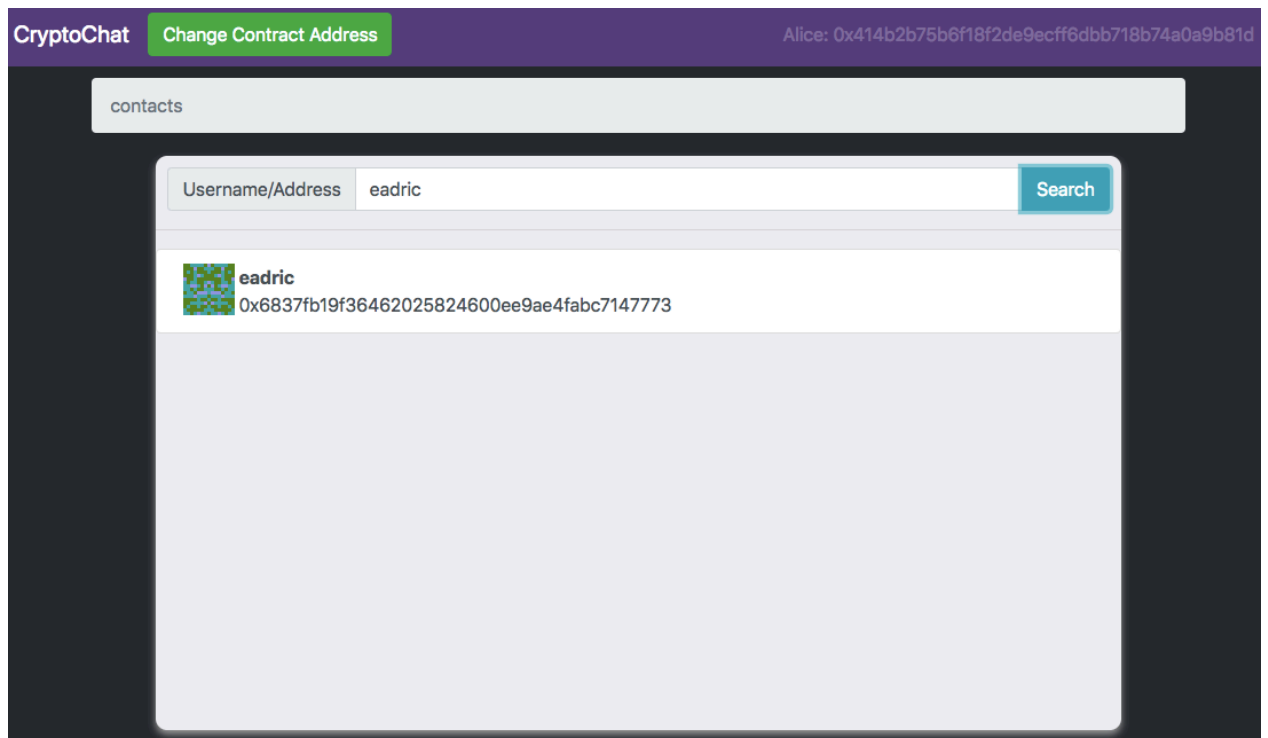
当用户进行注册或登录操作时，都会根据用户的信息进行 RSA 密钥的计算与验

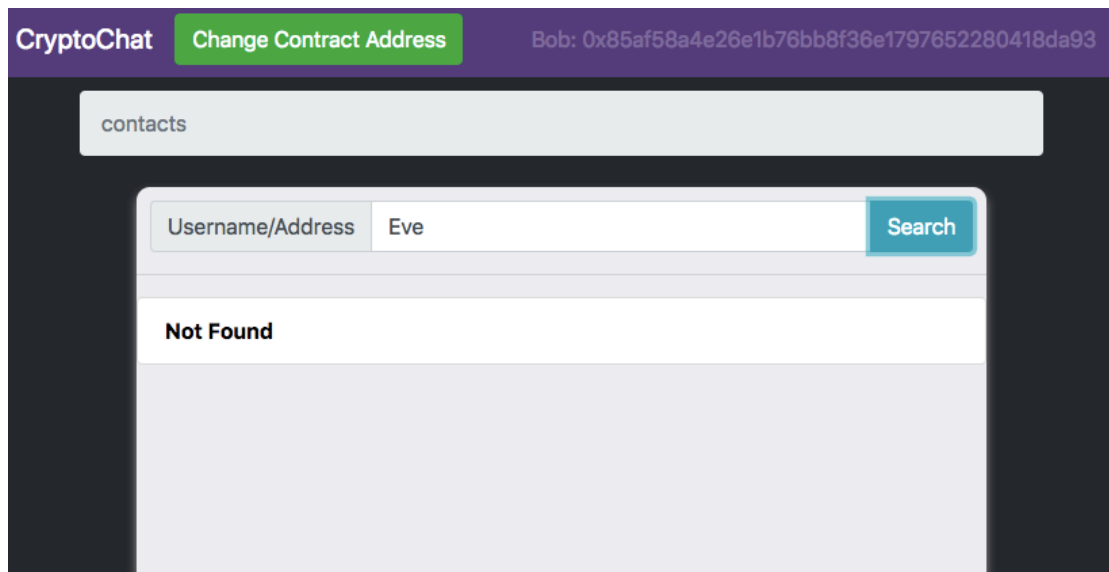
证，耗时可能较长，在 Chrome 浏览器中测试耗时约 10 秒，在其他浏览器下耗时为 30 秒至一分钟。注册过程中，若使用 Metamask，则会有两次弹窗提示，要求对用户信息签名以及请求调用注册的交易。

登录后进入联系人界面，该界面会显示所有曾经有过信息往来的用户：



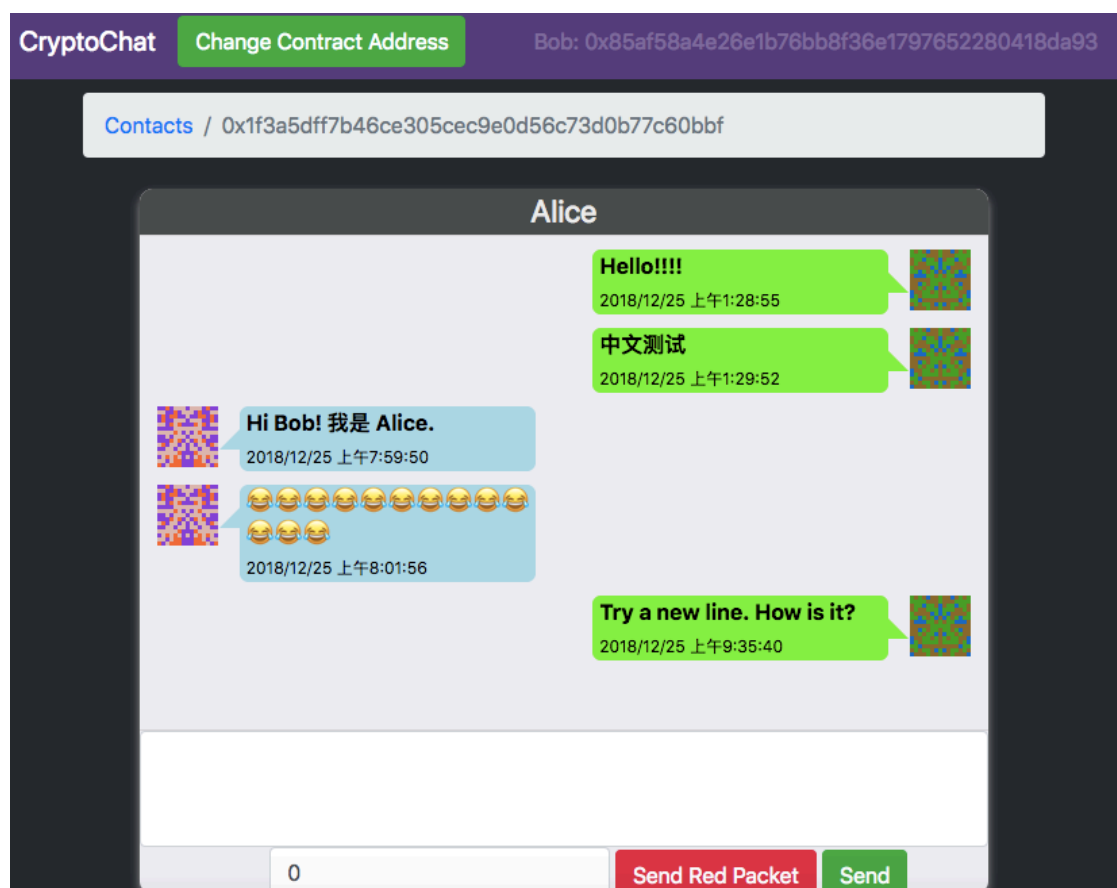
若要向其他用户发送信息，可从上方搜索栏查找用户名或地址，若用户存在则将显示用户，否则将显示 Not Found





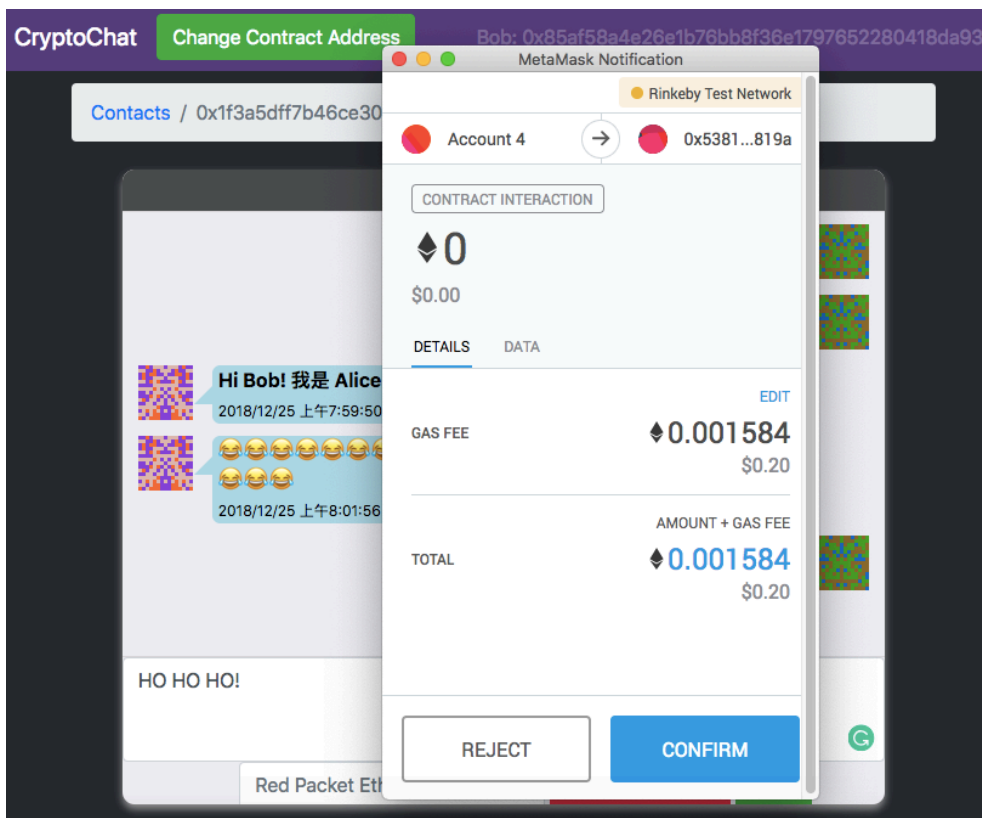
若要回到联系人界面，只需将搜索栏清空后点击搜索即可。

点击用户后进入聊天界面：

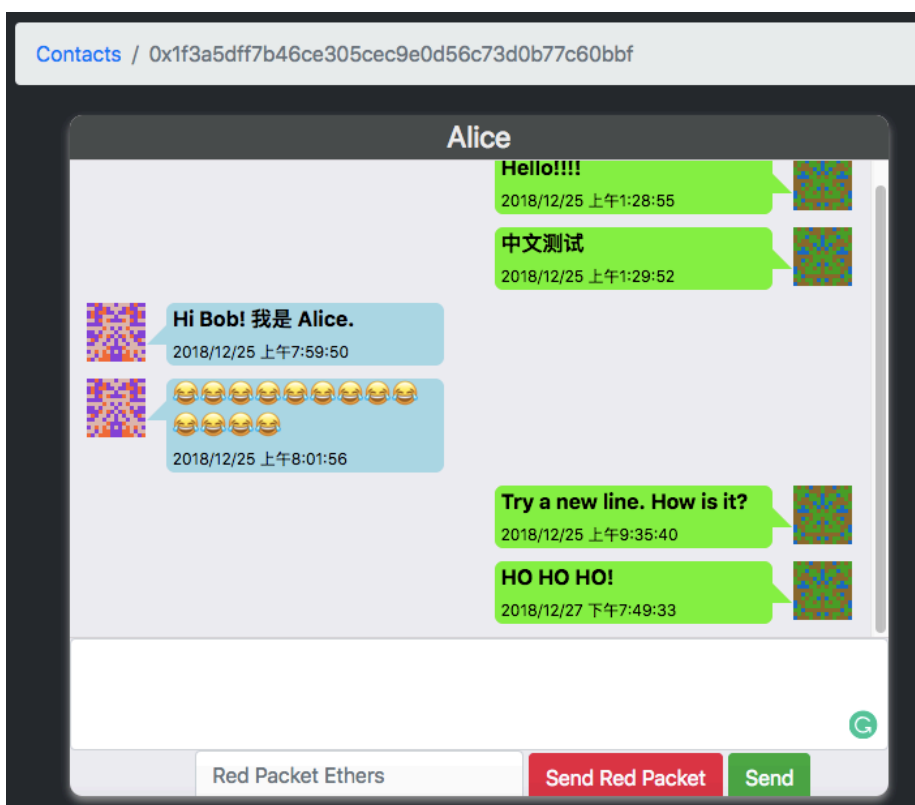


该界面下可进行消息与红包的发送，在文本输入框中输入消息内容点击发送

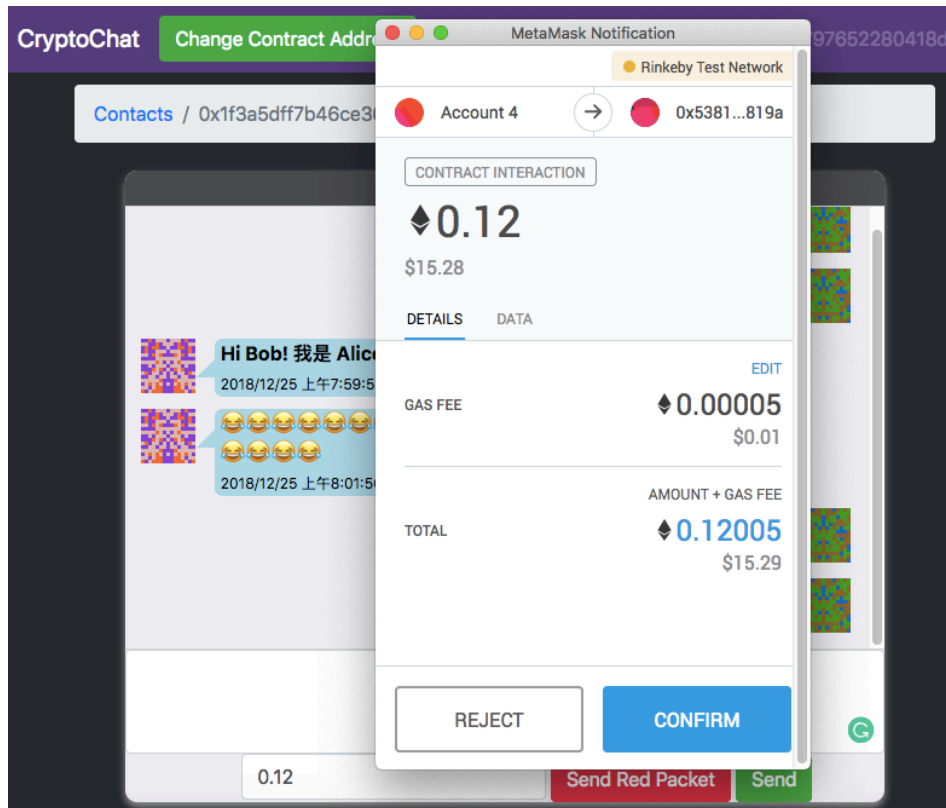
发出消息：



发送消息后需等待交易确认消息才能正确发出，应用前端将每 5 秒调用一次合约检查是否有新的发出或接受的消息。发送完成后将显示在界面上：



输入红包数额后点击发送红包即可发起红包的转账，该转账将向合约地址发送指定数量的以太币，合约将把所有接收的以太币转入指定的账户：

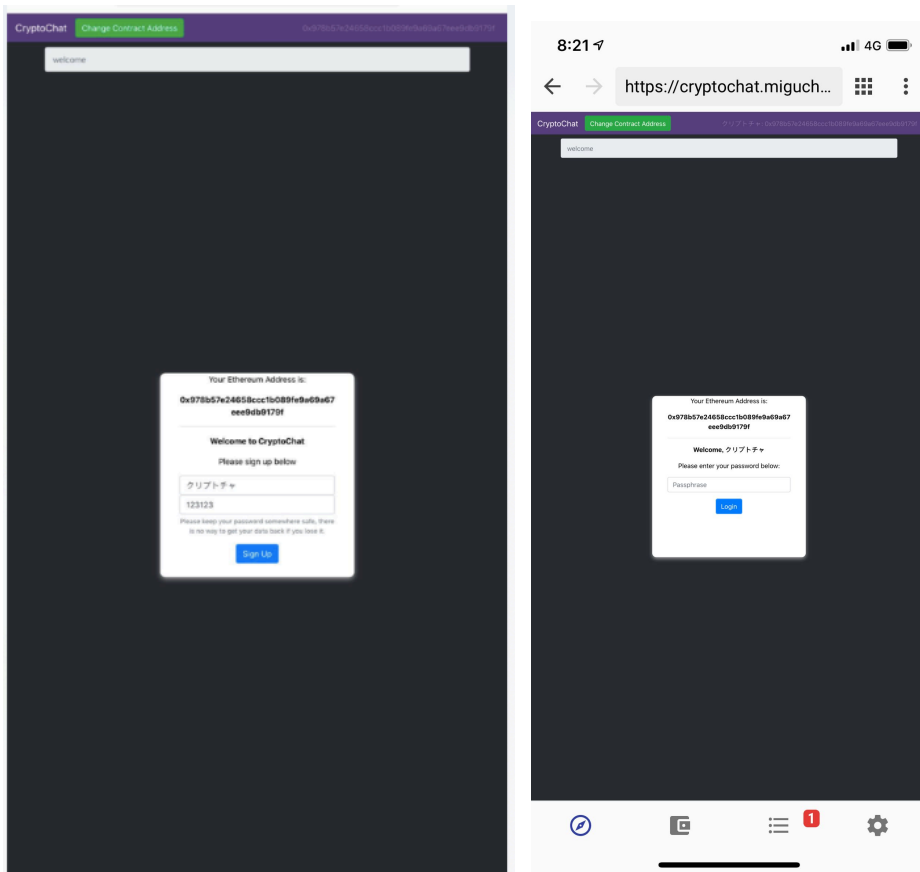


在聊天界面，点击左上角的 Contacts 连接即可返回联系人界面。

5. 测试

在 Firefox 与 Chrome 下进行应用运行的测试效果如上述使用方法过程中的截图相同，下面是在 iOS 中 Cipher 应用下运行的效果，由于界面没有对移动设备优化，界面的比例存在一点问题：

注册界面与登录界面：



注册过程中的签名与合约调用的请求：

8:19

4G

×


Sign Message

Signing messages may result in unexpected consequences. Only sign messages from sources that you trust.

Requester

https://cryptochat.miguch.com/

Signer Address


0x978B57e24658CcC1B089Fe9A69a67Ee9DB9179F

Message

Hex

UTF-8

```
{
  "pubKey": "MIGJAoGBAOG9Q9TA0jvkYnaDMzVbOqQmzTdauleibvc5rN4xXhuNJJThLdmA03OG1/VJevY3OxcO9UicuCXEpVjKXjypPGgmnL+WVm1ITZlVGacJBfEGwf85aQ5KXbpsYpDylweYK4eqNuZSLGoXB1fJYX9+XjKHOFsuQiFygeVD7LwdRagMBAAE=",
  "keySig": "f4E1JaA/WIBZav+YnZ6LVnicvrcDISEKMI+cA/tLKsychYApDc6u2skEZfmHnQqBYP2C65JtB+R6YgZZkOsfhEKmD1wnSe1zc/HB5mPWrvnKrFr4UhbXvzS4R4vWqHvTXRRI68wp+NogMZw2saLEbZJ9jLXje0iTuPssC8kk=",
  "username": "クリプトチャ",
  "address": "0x978b57e24658ccc1b089fe9a69a67eee9db9179f"
}
```

REJECT

APPROVE

8:19

4G


×

Sign Transaction


Requester

https://cryptochat.miguch.com/

Sender Address


0x978B57e24658CcC1B089Fe9A69a67Ee9DB9179F

Recipient Address


0x538132f00b5ccFf2a5bdd7b0d790307b55D2819A

Total Amount Including Estimated Gas Fee

Ξ0.000833887

≈ \$0.11

Included Data

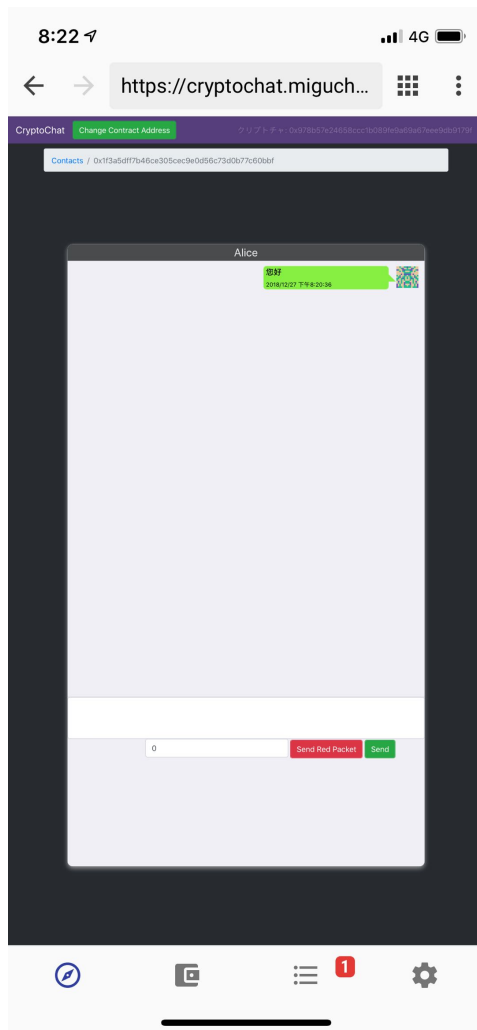
836 bytes

REVEAL

REJECT

APPROVE

聊天界面：



6. 项目中仍然存在的问题

在测试中发现的问题如下：

1. 登录注册时的耗时较长，在登录与注册时生成密钥的耗时较长，尤其在非 Chrome 浏览器上。
2. 程序中每 5 秒从区块链检查一次是否有新消息，当检查中出现问题时可能导致新消息无法显示，只能刷新页面后才能查看消息更新。
3. 在私链上进行测试时生成区块较快，发送的消息可以较快地被记录至合约数据中，但在 Ropsten 和 Rinkeby 测试网络下进行测试时发送消息的交易需要较长时间才能被确认，使得发送消息的延迟较大（或许本应用更适合被称为 CryptoMail）。

7. 本应用实现过程中使用的开源项目

- [ethereum/blockies](#): 根据地址生成用户头像。
- [ethjs/ethjs](#): 使得程序Web端可以用Promise的方法异步地与区块链进行交互，而不需要通过 Metamask 的回调函数，避免了回调地狱。
- [golang/crypto](#) : 使用 Go 进行 RSA 和 AES 的数据加解密以及数字签名。
- [gopherjs/gopherjs](#): 将 Go 编译为 JavaScript 供程序使用。
- [vuejs/vue](#) 前端页面使用的框架。
- [bootstrap-vue/bootstrap-vue](#) 程序界面所使用的Bootstrap样式组件。