

iAuth 身份认证与账户信息存储的智能合约及应用

一、 选题背景

在大数据浪潮下，没有什么数据不能拿来研究，当然包含大量用户信息的账户数据首当其冲；另一方面，随着网络服务不断增加，用户需要注册大量账号来使用或享受这些服务——作为用户身份识别，即使有一些网站提供已有账户登录的服务下（例如，你可以用 Google 账号登录 Github）。在这样的背景下，对于第一点，账户需要有一定的匿名性，同时账户最好可以掌握在自己手中，而不是存储在某个数据库或机构中；对于第二点，账户需要有泛用性；而这些正是区块链与智能合约的特点，去中心化存储、匿名性、泛用性，因此编写了一个这样的智能合约的应用。

二、 应用功能

1. 用户与网站使用该合约后，用户可以用其区块链账户登录网站，网站在区块链上验证账户归属。这认证属于零知识证明，用户将公钥放置到区块链供网站查询，网站给予用户挑战，用户用私钥将挑战加密后写入区块链，之后网站可以获取到加密后的挑战，用公钥解开来验证是否为原挑战，以此来验证用户身份。
2. 用户可以选择关闭合约的使用，防止信息的泄露，当然也无法继续登录了
3. 用户可以设置昵称，用于网站名称的显示
4. 用户可以设置详细信息，用于网站的其他功能(如邮箱私信)，这里可以设置的是邮箱和电话
5. 用户可以关闭获取详细信息的权限，这样网站就无法获取得到
6. 一个用户可能拥有多个账户，不同账户的详细信息不用分别设置供网站获取，只需要同步某个账户的详细信息即可，当然，这里要验证你是否为该账户的主人，用同样的零知识证明的方法来验证

三、 Github 源码地址

<https://github.com/Alicizations/iAuth>

四、 使用说明

(1) 环境要求

NodeJS	v8.9+
Npm	v5.6+
Chrome	v70+
MetaMask	v5.2.2+
Ganache	v1.2.2+

(2) 启动项目

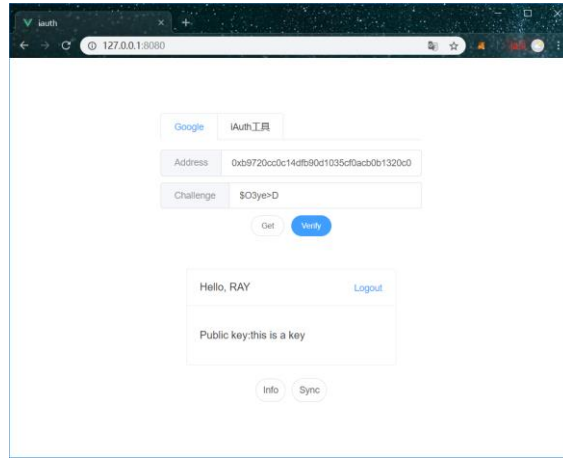
先 clone Github 上的项目到 iAuth 文件夹
在 iAuth 文件夹中打开命令行:

```
npm install
```

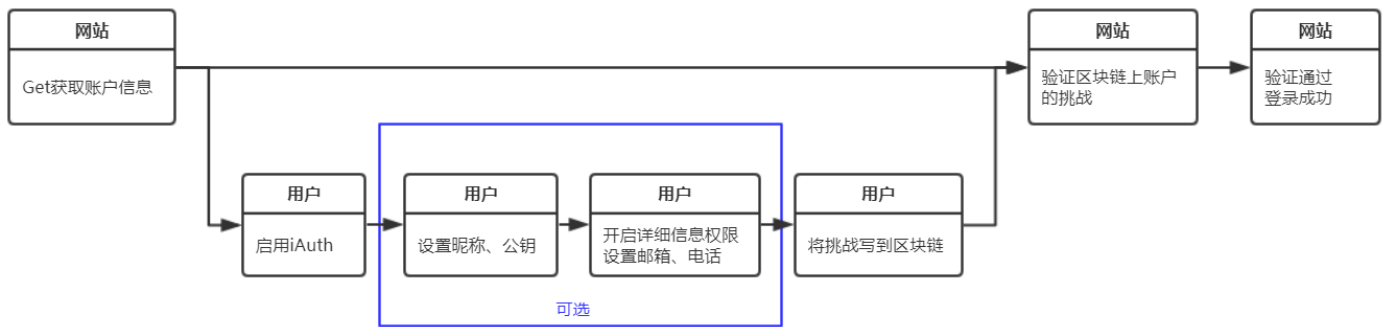
安装完后运行:

```
npm run serve
```

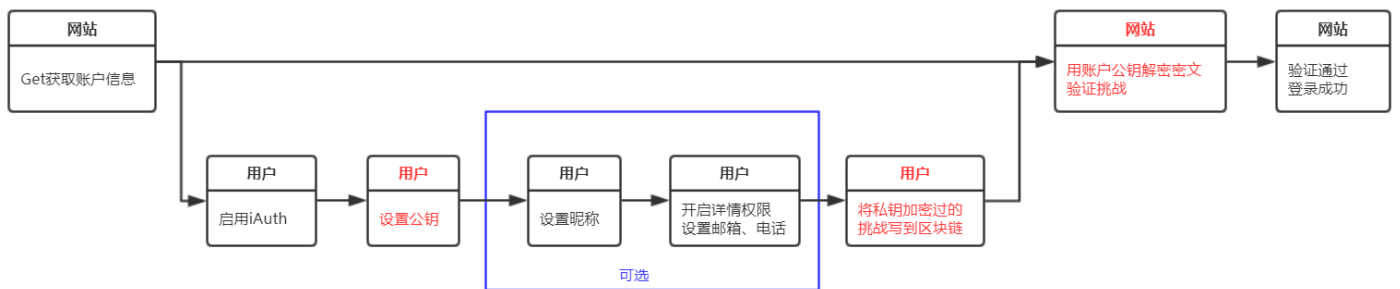
(3) 网页说明



网页界面



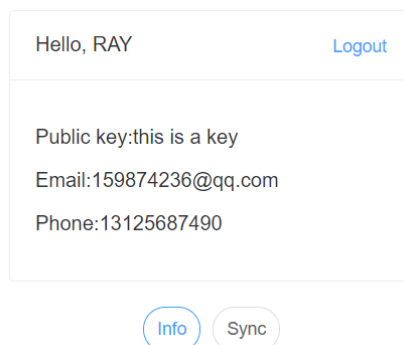
弱验证的登录流程



强验证的登录流程



模拟要登录的网站 && 配置账户 iAuth 智能合约的工具



成功登录后的信息

控制台

nikename

控制台

email

启用iAuth

启用Extra

基本信息

额外信息

验证密文

public key

phone

Send

启用 iAuth 开启详细信息权限 && 设置昵称、公钥 && 设置邮箱和电话

Google

iAuth工具

控制台

\$03ye>D

基本信息

private key

额外信息

验证密文

Send

将挑战经过私钥加密后写入区块链

见图中

五、 测试

Ganache

ACCOUNTS BLOCKS TRANSACTIONS LOGS

SEARCH FOR BLOCK NUMBERS OR TX HASHES

CURRENT BLOCK

GAS PRICE

GAS LIMIT

NETWORK ID

RPC SERVER

MINING STATUS

MNEMONIC

weird powder peanut timber humor acid tooth couch release bird organ

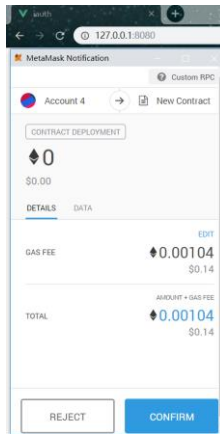
HD PATH

m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX	
0xb0E331546414f9C3Ad8D8b98A5CAD34AbdfED6Ea	100.00 ETH	0	0	
0xE9ef95f17C46a5c8aB01134CD2800adBC9F921Bd	100.00 ETH	0	1	
0x1B348eCf179A079b04a2a9A7479c34540EE58B28	100.00 ETH	0	2	

打开 Ganache

打开 Chrome 登录 MetaMask, 连接到 Ganache 网络, 导入前两个账户



打开 <http://127.0.0.1:8080>, 稍等一会既进行部署 iAuth 合约

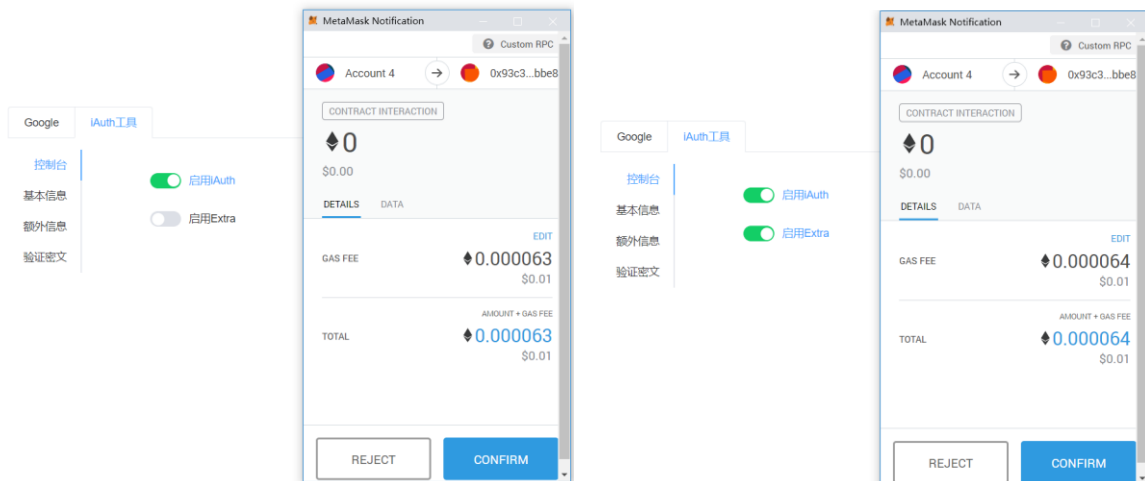
[Google](#)
[iAuth工具](#)

Address

Challenge

[Get](#)
[Verify](#)

点击 Get, 获取当前账户和挑战(用户接下来按照使用说明进行合约的设置)



启用 iAuth 和详细信息



[Google](#)
[iAuth工具](#)

控制台

BlockChainer

```
-----BEGIN PUBLIC KEY-----
MFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBAGvY
659001cW0xhu8I2rDcwfoeC4S6A
pTfcbJluyET/ND0HmZKwvZY/87iASKa9e
uluCrSTlkkeGnGciMCgEAQ==
-----END PUBLIC KEY-----
```

[Send](#)

[Google](#)
[iAuth工具](#)

控制台

[Send](#)

设置昵称、公钥 && 邮箱、电话(交易确认图略)

Google

iAuth工具

Address

0x4fec588b42fd1c6b34418d869b8899d923f

Challenge

@13vk^d6CP<

Get

Verify

切换到另一个账户

Google

iAuth工具

Address

0x4fec588b42fd1c6b34418d869b8899d923f

Challenge

@13vk^d6CP<

Get

Verify

Hello, test1111

Logout

Public key:NO SETTING

Email:NO SETTING

Phone:NO SETTING

Info

Sync

配置并登录这个账户，可以看到只设置了昵称，其他都是空的

Sync extra information

×

0xe4e18a83d2909993a90d1708b9939a46b8f136d5

Z\$)z!K1D

verify

进行账户详细信息同步

✔ Sync successfully

Google

iAuth工具

Address

0x4fec588b42fd1c6b34418d869b8899d923f

Challenge

@13vk^d6CP<

Get

Verify

Hello, test1111

Logout

Public key:NO SETTING

Email:951234768@qq.com

Phone:13125896347

Info

Sync

同步成功，可以看到邮箱和电话信息更新

六、 小结

一开始也是想做身份认证这方面的，在网上看到了又类似的想法，就实现了一个有这样功能的智能合约，实现得非常简陋吧，网页开发用了 Vue，感觉不是很必要…只不过当时写了一半了，只能硬着头皮写下去了。然后经过课程+项目也算是对区块链有个入门的了解了吧，但是感觉其具体实现与各种知识还是很多需要去学习的，总之最后能用区块链做一个这样的小项目还是挺有收获的。