钱包模块：钱包分类

虚拟货币钱包实际上是管理和储存 key（私钥）的工具，而帐户是从公钥衍生出来。主要有三种类型：轻钱包模式、重钱包模式和兼容模式。

轻钱包模式：

轻钱包模式下，需要有一个开放的节点与钱包通信,可能是Http RPC协议，这个节点可是任意链上的节点。轻钱包通常做成浏览器插件，插件在运行时会自动注入Web3框架，DApp可以通过Web3与区块链节点通信。当DApp只是单纯的获取数据时是不需要钱包介入的，但是当DApp需要发送交易到链上时需要通过钱包完成对交易签名的过程。优点：不需要用户同步区块链节点就可使用
缺点：需要一个公开的节点提供服务，可能会存在安全性问题

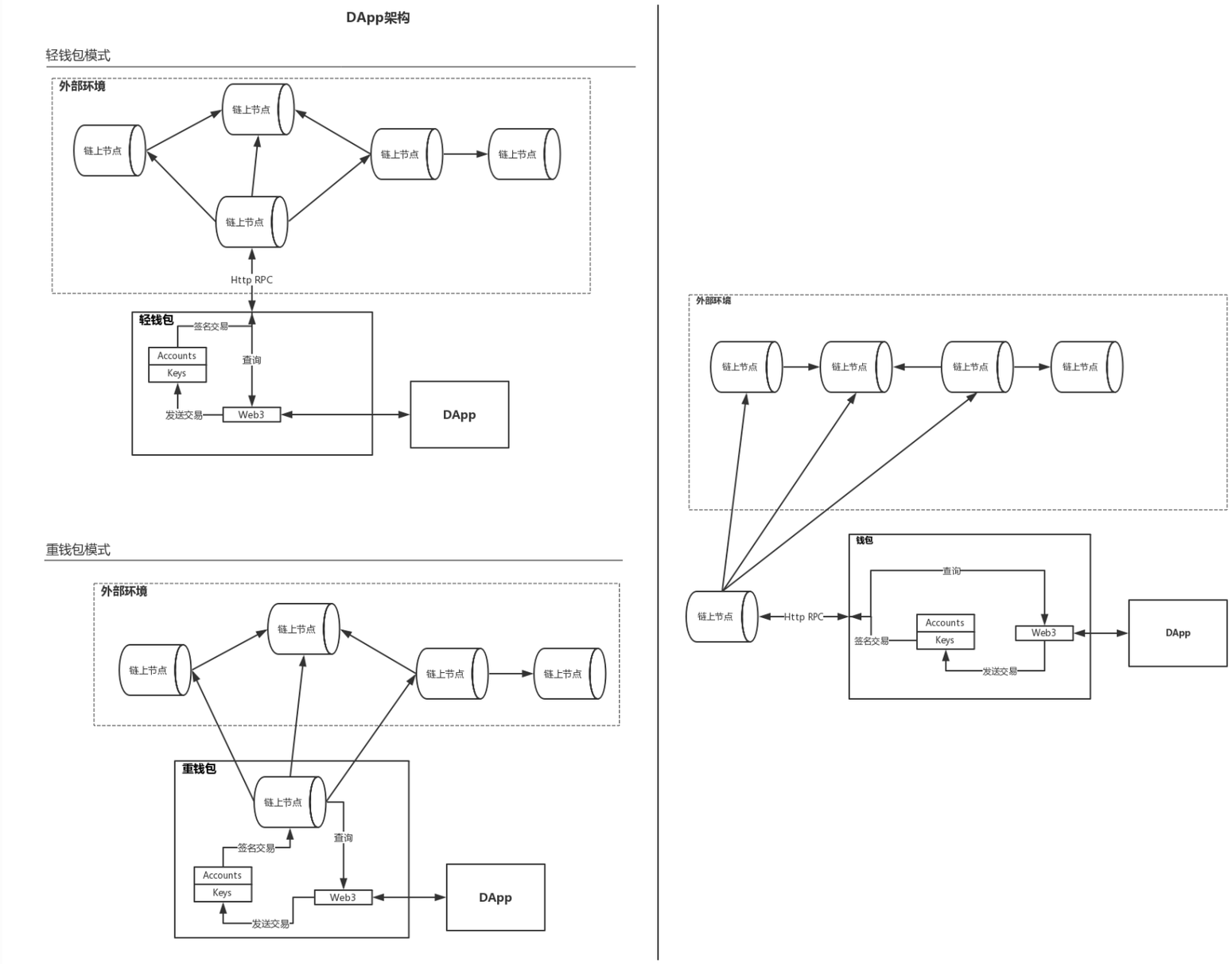
重钱包模式：

重钱包会自己同步并持有一个区块链节点，提供一个浏览器环境，其他与钱包相似。优点：自己持有并同步节点，安全性高缺点：需要持有一个全量的区块链节点

兼容模式：

兼容模式可以在轻钱包和重钱包下同时使用，与钱包通信的节点可以选择在钱包外本地持有，也可以自己搭建服务持有并公布节点。

钱包模块：钱包分类


DApp架构
轻钱包模式
外部环境
链上节点
链上节点
链上节点
链上节点
链上节点
Http RPC
轻钱包
签名交易
Accounts
Keys
查询
发送交易
Web3
DApp
重钱包模式
外部环境
链上节点
链上节点
链上节点
链上节点
链上节点
重钱包
链上节点
签名交易
查询
Accounts
Keys
Web3
发送交易
DApp
外部环境
链上节点
链上节点
链上节点
链上节点
链上节点
Http RPC
钱包
查询
Accounts
Keys
Web3
签名交易
发送交易
DApp

钱包模块：钱包主要功能

基本功能:

- Send
- Receive
- Smart Contract
- Transactions (This is a more detailed and full historical set of transactions)
- Notification

金额显示

- GetAvailable
- GetPending
- GetTotal
- GetGasPrice
- GetAll

用户功能:

- FromRandNumber
- FromPrivateKey
- FromMnemonic + path
- GetAddress
- GetICAPAddress
- GetChkSumAddress
- GetPublicKey
- GetPrivateKey

管理功能:

- BackupKeys
- BackupTransactions
- BackupBalances
- GetHistoryTransactions
- SearchHistoryTranstion
- GetBlockNumber

钱包模块：钱包安全（备份）

钱包的形态多样，备份的方式多中，根本目的： 防盗，防丢，分散风险。主要有：

多处和分离备份 keystore && password：

- 将 keystore 文件放置多处安全的位置，如离线的 USB 以及你信任的云存储服务商。
- keystone 对应的 password，你应该采用强密码，同样多处且与 keystore 分离备份。

纸钱包备份：

- 纸钱包实质就是将 keystore 或 私钥以纸质化形式保存，一般为二维码形式，直接打印对应的二维码纸钱包。

脑钱包：

- 通过 BIP 39 提案的方式生成足够随机的，可记忆的助记码。

多重签名

- 多重签名是一个不错的选择，它的优势是当你需要提取超过限制的金额时，需要多把私钥同时授权，同时提升防盗，防丢的安全性。

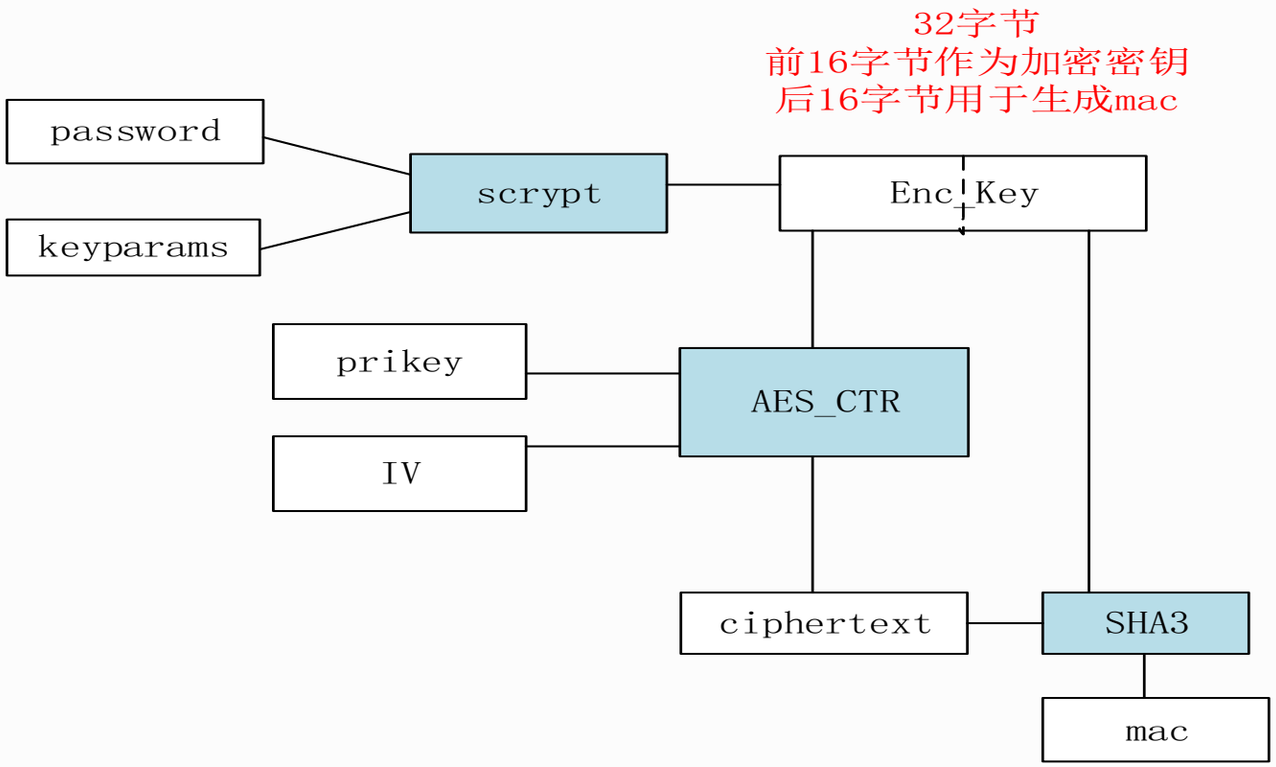
钱包模块：账户文件说明

Keystore目录账户文件格式： 一般 UTC--<created_at UTC ISO8601>-<address hex>， 如： UTC--2018-07-02T03-56-34.619039938Z--f8ffd54afc5a41b352af2feec9cc447b90a4cfbb

```
008aeeda4d805471df9b2a5b0f38a0c3bcba786b   地址
{
    "crypto" : {
        "cipher" : "aes-128-ctr",   加密私钥的算法
        "cipherparams" : {          加密需要使用的参数
            "iv" : "83dbcc02d8ccb40e466191a123791e0e"
        },
        "ciphertext" :              私钥加密后的密文
"d172bf743a674da9cdad04534d56926ef8358534d458fffccd4e6ad2fbde479c",
        "kdf" : "scrypt",           密钥生成函数使用的算法
        "kdfparams" : {             scrypt函数使用的参数
            "dklen" : 32,
            "n" : 262144,
            "r" : 1,
            "p" : 8,
            "salt" :
"ab0c7876052600dd703518d6fc3fe8984592145b591fc8fb5c6d43190334ba19"
        },
        "mac" :                     用于校验正确性
"2103ac29920d71da29f15d75b4a16dbe95cfd7ff8faea1056c33131d846e3097"
    },
    "id" : "3198bc9c-6672-5ab3-d995-4942343ae5b6",
    "version" : 3
}
```

- cipher：对称算法的名称，该算法用于加密私钥
- cipherparams：上述 cipher 算法需要的参数;
- ciphertext：以太坊私钥使用上述 cipher 算法进行加密的结果
- kdf：密钥生成函数，用密码生成加密 keystore 文件的密钥
- kdfparams：上述 kdf 算法需要的参数;
- Mac：用于验证密码的正确性。

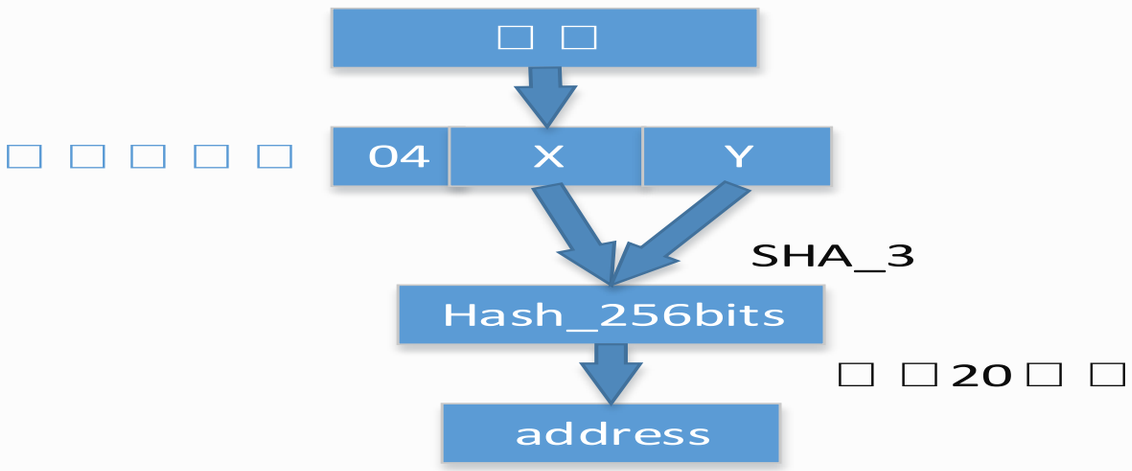
钱包模块：私钥和账户地址生成



① 首先，根据密码和参数keyparams经过scrypt函数来计算加密密钥。

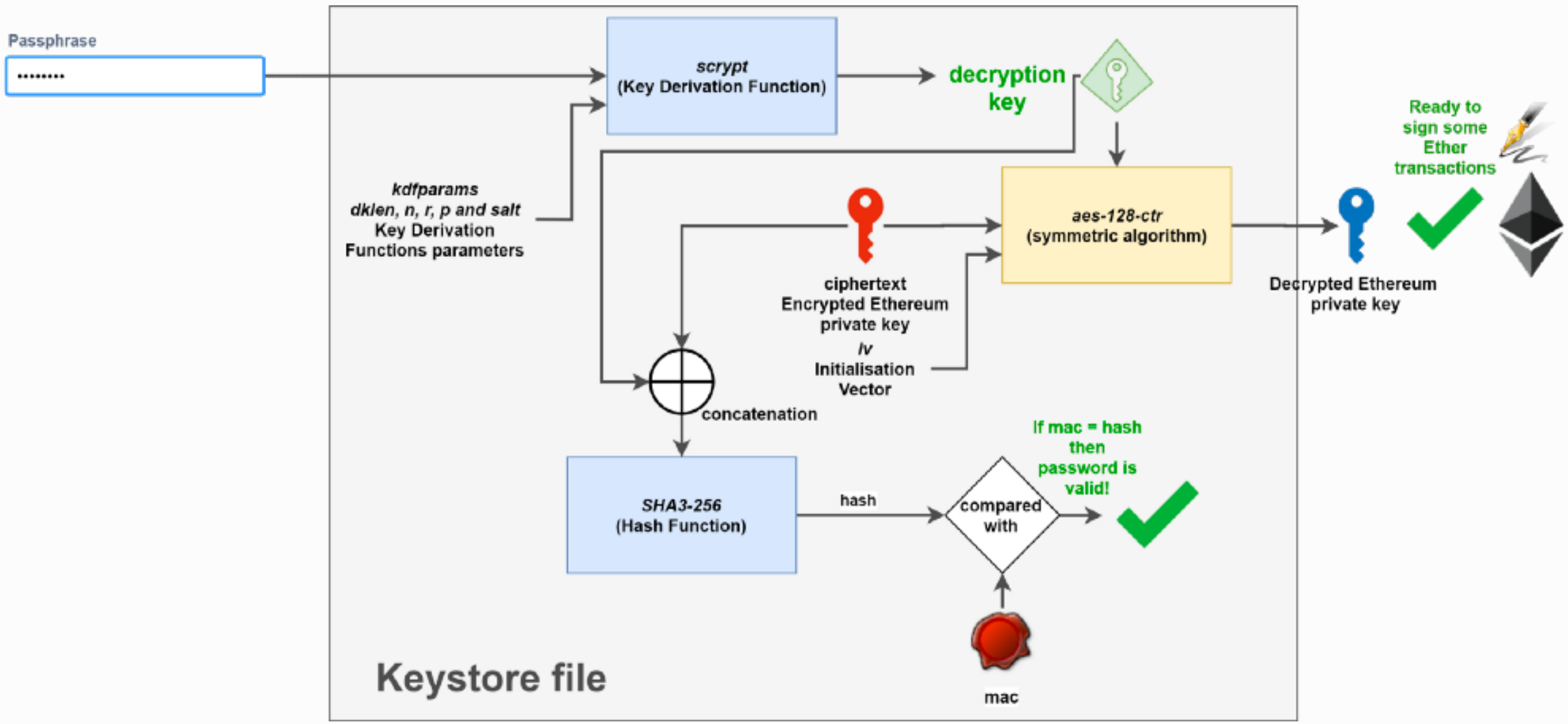
② 然后，用计算出的加密密钥前16字节和参数IV对prikey使用AES_CTR进行加密，得到私钥的密文ciphertext。

③ 最后，将加密密钥后16字节和ciphertext 密文一起用SHA3计算得到mac。



1. 首先，上图中的prikey生成public key。
2. 然后使用Keccak256算法对public key计算hash，取后20字节作为address

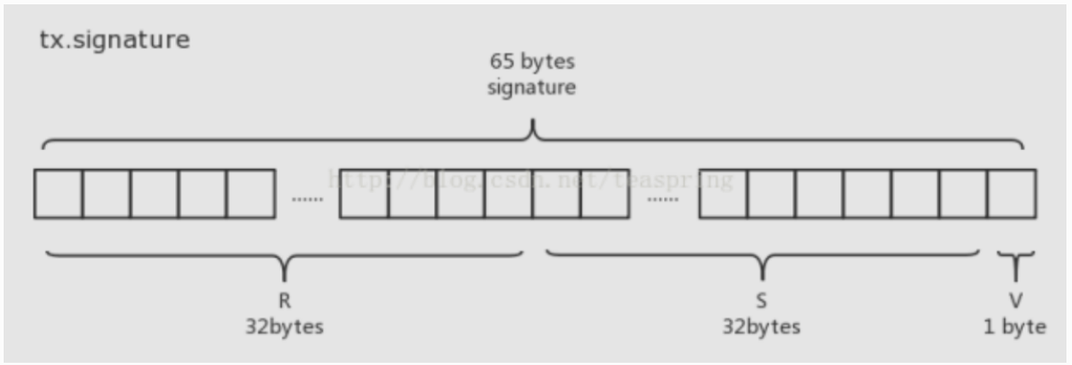
钱包模块：秘钥解密



首先，输入密码，这个密码作为 kdf 密钥生成函数的输入，来计算解密密钥。
然后，计算出的解密密钥和 ciphertext 密文连接做SHA_3，和 mac 比较来确保密码是正确的。
最后，用解密密钥对 ciphertext 密文解密
其实就是读取密钥文件和加密密码，对私钥进行解密，以便使用私钥对发送的交易进行签名

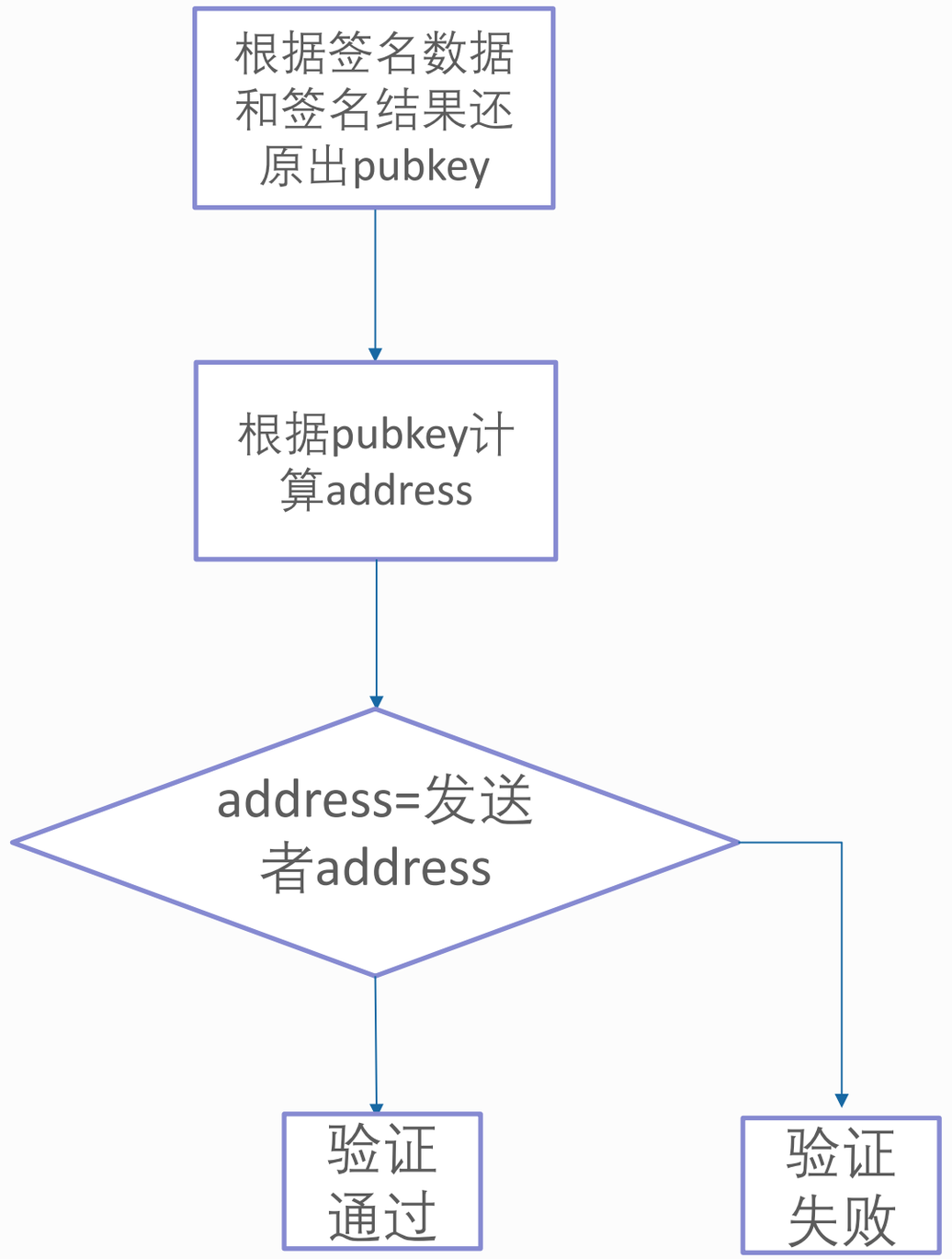
钱包：交易签名和验证

以太坊的交易使用私钥来签名。因此发起交易之前必须要进行对钱包的解锁操作，得到私钥。
签名结果的结构如下：



其中前64字节为签名值。最后一字节为标志位（有效标志位为0,1），用于签名验证时还原公钥。签名算法步骤及标志位的标志方法如下：

- 输入：椭圆曲线参数(q, FR, S, a, b, P, n, h)；私钥d；消息的哈希e=hash(m)
- 输出：签名结果(r, s, v)



钱包：附录之BIP和多账户

见：http://10.1.2.46:8090/pages/viewpage.action?pageId=11403632

钱包模块：ICAP(Inter exchange Client Address Protocol)

ICAP 互换客户端地址协议，一种IBAN兼容系统，用于引用和处理客户帐户，旨在简化资金转移流程。

在第三方账户之间（特别是交易所账户）之间转账资金给用户带来了相当大的负担，并且由于客户账户中的存款被识别的方式而容易出错。现有的银行业通过拥有一个被称为IBAN的通用代码解决了这个问题。该代码合并了机构和客户帐户以及错误检测机制，实际上消除了微不足道的错误并为用户提供了相当大的便利。

国际银行账户号码~[1]~ (International Bank Account Number，简称IBAN)是各国各银行之间互相定立的标识号码，可降低国际间金融操作的失误。它最初是由欧洲银行标准委员会（ECBS）通过，后来被采纳为国际标准 ISO 13616:1997。目前的标准是ISO 13616:2007，表明SWIFT代码（ISO 9362）为正式的格式。最初开发是为了促进欧盟范围内的支付，但现在也已经实施到大多数欧洲国家和其他国家，尤其是在中东和加勒比海地区。

钱包模块：ICAP(Inter exchange Client Address Protocol)

IBAN结构组成：IBAN代码由多达34个不区分大小写的字母数字字符组成，其字符取值范围为0-9和A-Z。它包含三个信息：

- 国家代码; 以下内容的顶级标识符（ISO 3166-1 alpha-2）；
- 错误检测代码; 使用mod-97-10校验和协议（ISO / IEC 7064：2003）；
- 基本银行账号（BBAN）；该机构，分支和客户账户的标识符，其组成取决于上述

国家	格式	说明
英国🇬🇧	`GBkk bbbb ssss sscc cccc cc`	b = 银行代码 s = 银行分类代码 c = 账号

该 `GBkk bbbb ssss sscc cccc cc` 格式解读为：

`[国家代码:GB][错误校验码:kk][基本银行账号:bbbb ssss sscc cccc cc]`

对于英国来说，BBAN由以下部分组成：

- 机构标识符，4个字符的字母，例如 `MIDL` 代表汇丰银行。
- 分类代码（机构内的分行标识符），一个6位十进制数字，例如`402702`汇丰银行的Lancaster分行。
- 帐号（分支机构内的客户标识符），一个8位十进制数字。

钱包模块：ICAP(Inter exchange Client Address Protocol)

以太坊引入了新的IBAN国家代码：XE，X前缀为扩展的意思，E表示为以太坊（Ethereum），制定为非法定货币（例如XBOX, XEOS）

■ 直接类型：

- 此代码的直接BBAN为30个字符，将包含一个字段：
- 帐户标识符，30个字母数字（<155位）。这将被解释为表示160位以太坊地址的最低有效位的大端编码的36进制整数。因此，这些以太坊地址通常以零字节开始。

■ 基本类型：

- 与直接编码相同，但代码为31个字符（与IBAN不兼容）并组成相同的单个字段：
- 帐户标识符，31个字符字母数字（<161位）。这将被解释为一个代表160位以太坊地址的大端编码的36进制整数。

■ 间接类型：

- 该代码的BBAN在间接时将为16个字符，并将包含三个字段：
- 资产标识符，3个字符的字母数字（<16位）；
- 机构标识符，4个字符的字母数字（<21位）；
- 机构客户标识符，9个字符的字母数字（<47位）；
- 包括四个首字符，这导致最终的客户帐户地址长度为20个字符如：*XE81ETHXREGGAVOFYORK*
- Uri表示为 *iban:XE7338O073KYGTWWZN0F2WZ0R8PX5ZPPZS*

钱包：附录之ABI

ABI(Application Binary Interface)：是强类型的，在编译时和静态时都知道的，它是与以太坊系统contract与contract的交互标准方式。

数据类型：

1. uint<M>: unsigned integer type of M bits, 0 < M <= 256, M % 8 == 0. e.g. uint32, uint8, uint256.
2. int<M>: two’s complement signed integer type of M bits, 0 < M <= 256, M % 8 == 0.
3. address: equivalent to uint160, except for the assumed interpretation and language typing. For computing the function selector, address is used.
4. uint, int: synonyms for uint256, int256 respectively. For computing the function selector, uint256 and int256 have to be used.
5. bool: equivalent to uint8 restricted to the values 0 and 1. For computing the function selector, bool is used.
6. fixed<M>x<N>: signed fixed-point decimal number of M bits, 8 <= M <= 256, M % 8 ==0, and 0 < N <= 80, which denotes the value v as v / (10 ** N).
7. ufixed<M>x<N>: unsigned variant of fixed<M>x<N>.
8. fixed, ufixed: synonyms for fixed128x18, ufixed128x18 respectively. For computing the function selector, fixed128x18 and ufixed128x18 have to be used.
9. bytes<M>: binary type of M bytes, 0 < M <= 32.
10. function: an address (20 bytes) followed by a function selector (4 bytes). Encoded identical to bytes24.

钱包模块：ABI

The following (fixed-size) array type exists:

- <type>[M]: a fixed-length array of M elements, M >= 0, of the given type.

The following non-fixed-size types exist:

- bytes: dynamic sized byte sequence.
- string: dynamic sized unicode string assumed to be UTF-8 encoded.
- <type>[]: a variable-length array of elements of the given type.