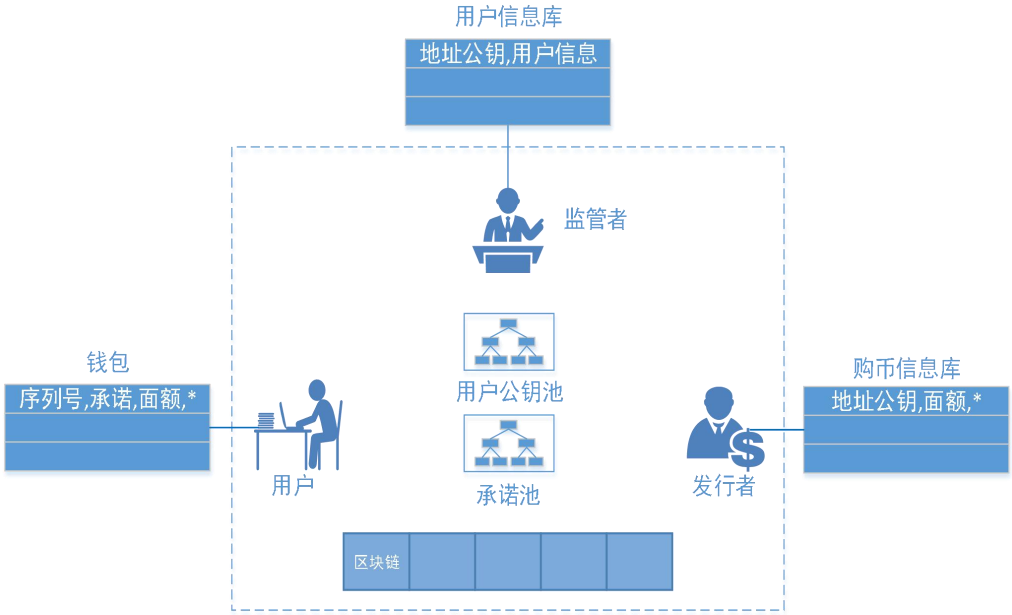
系统模型



系统模型图

说明：

1. 监管者：监管整个系统的所有交易，可以看清每笔交易的金额和流向。
2. 发行者：负责向用户发行数字货币，为系统的可信方。
3. 用户：系统中进行数字货币交易的用户。
4. 用户信息库：保存系统中所有用户的信息，由监管者独立维护，由记录$<U_{pk},*>$构成，其中U_{pk}为用户的地址公钥，*为用户其它信息（根据业务需要）。
5. 购币信息库：保存所有用户购买数字货币的信息（假设系统中用户是从发行者购买数字货币后，再进行交易的），由发行者自己维护，由记录$<U_{pk}, v, *>$构成，其中U_{pk}为用户的地址公钥，v 为购买的面额，*为用户其它信息（根据业务需要）。
6. 钱包：保存用户自己拥有的数字货币，由用户自己独立维护，由记录$<v, SN, CM, \rho, *>$构成， v 为面额，SN 为序列号，CM 为相应的承诺，ρ 为随机数，*为其它信息（根据业务需要）。
7. 用户公钥池：存储系统中当前所有合法用户公钥的哈希。

8. 承诺池：存储系统中的所有代币承诺的哈希。
9. 区块链：存储系统的交易信息，以及承诺池和用户公钥池的哈希值。

方案流程

1. 参数初始化：

这里的参数是指后续流程中监管者的 Elgamal 加密公钥，由监管者 Q 生成。

1. 监管者 Q 选择一个 q 阶循环群 G；
2. 监管者 Q 选择群 G 的两个生成元 G1，G2；
3. 监管者 Q 选择一个随机数 x 并保留为自己的私钥，计算 H = x * G2；
4. 监管者 Q 公布 Elgamal 公钥 Enc_pk=（G1,G2,H）。

2. 注册：

用户必须注册才能进行后续的购币和交易操作，用户注册的核心操作：监管者将用户的地址公钥哈希值存储到用户公钥池中。监管者维护一个公钥池。

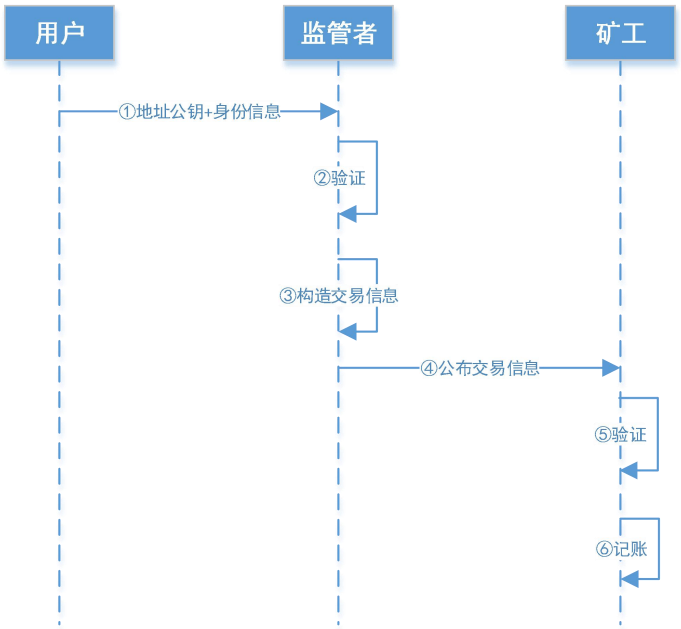


图 2 用户注册流程图

说明：

（1） 步骤①-②可以是线下的操作或者结合其他系统来进行，其核心工作是监管者在用户信息库中记录下用户的地址公钥和其他个人信息。

（2） 步骤③监管者选择随机数 ρ，并计算用户公钥的哈希值H_{pk}，然后签名$\sigma_G = Sig_G(\rho, H_{pk})$，然后构造注册交易信息：

MSG_{reg}=(ID_{Tx}， $Type_{Tx}$ ，σ_G， ρ， H_{pk})，其中ID_{Tx}为交易标识，$Type_{Tx}$为交易类型。

（3） 步骤④中监管者公布注册交易信息MSG_{reg}。

（4） 步骤⑤中，矿工验证签名σ_G，是否已经在链上，如果在链上，则认为该签名无效，然后验证签名的正确性，验证通过则进行记账。流程如下：

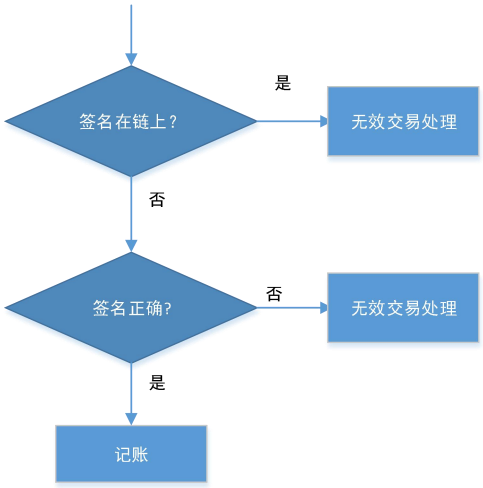


图 10 矿工验证

（5） 步骤⑥中，矿工将用户公钥的哈希值H_{pk}加入用户公钥哈希树，并重新计算各个节点的哈希值，然后将本次交易信息和用户公钥哈希树的根节点值一起写入区块中。

3. 购币：

假设用户是通过向发行者购买的方式获得初始数字货币的，然后才能进行转账交易。购币的核心操作：发行者将用户购得的数字货币承诺的哈希值添加到系统的承诺池中。其中承诺 CM 形式为 Pederson 承诺。

1. 用户想购买面额为 v 的代币，则向发行者提交购币请求，其中包含自己的地址公钥 pk 与购买资金 v。

2. 发行者首先在用户公钥池中查询该地址公钥，若该地址公钥存在于公钥池中，则发行者计算金额为 v 的被购代币的承诺 CM_v = v * G1 + r * H 和其中的 r。(或者用户自己生成 CM_v，将 CM_v 和随机数 r 都发给发行者，发行者验证 CM_v 成功后进行下一步)

3. 发行者对本次购币进行签名，即计算 SIG = Sig(ID, v)。其中，ID 为本次购币交易的标识。

4. 使用监管者公钥加密本次购买用户的购币内容，即计算 E(pk,v)。

5. 发行者生成并公布需要上链的购币交易信息 MSG_buy = (ID, SIG, CM_v, E(pk,v))。

6. 矿工验证购币交易信息 MSG_buy，若验证成功，则将承诺 CM_v 加入承诺池，即承诺 Merkle 树。

7. 发行者向用户返回 CM_v 和其中的随机数 r，用户使用 v 和 r 验证 CM_v 的正确性，确定发行者确实给自己返回了一个面额为 v 的代币承诺。

8. 用户使用自己的地址私钥 sk 和随机数 r，为购买得到的代币生成序列号 SN = H（r, sk)。

9. 用户将 CM_v、SN、v、r 存入自己的钱包。

4. 交易：

假设交易前发送者 S 与接收者 R 已协商好交易金额 v_r 与接收者 R 的地址公钥 R_pk，且接收者 R 使用步骤一中的公钥生成算法生成本次交易中 R 的公钥 (G1_R, G2_R, H_R)与私钥 x_R。现在，用户 S 要向用户 R 支付 v_r。S 原本有 v_o，支付 v_r，给自己找零 v_s。

交易时，

1. 用户 S 选择随机数 r_s1、r_r1，对交易金额 v_r 和找零金额 v_s 生成承诺：

$$CM\_s = v\_s * G1 + r\_s1 * H,$$

$$CM\_r = v\_r * G1 + r\_r1 * H,$$

其中对原始金额 v_o 的承诺是购币时就已生成的，CM_o = v_o * G1 + r_o1 * H。

2. 用户 S 选择随机数 r_r2 加密金额 v_r，使用接收者 R 的 Elgamal 公钥加密得到: E(v_r) = (v_r * G1_R + r_r2 * H_R, r_r2 * G2_R)。

3. 用户 S 选择随机数 r_spk、r_rpk，r_r3 分别加密交易双方地址公钥和交易金额 v_r，使用监管者 Q 的 Elgamal 公钥加密得到: E(v_r)' = (v_r * G1 + r_r3 * H, r_r3 * G2; S_pk * G1 + r_spk * H，r_spk * G2; R_pk * G1 + r_rpk * H，r_rpk * G2)。

4. 用户 S 生成零知识证明，证明：

（1） 用户 S 的原始货币 v_o 存在：验证承诺 CM_o 是否存在于承诺池中。

（2） 所有承诺和密文的格式正确：使用零知识**格式正确证明**。

（3） 对承诺 CM_o，CM_s，CM_r 使用 **range proof**，证明原始金额、交易金额和找零金额都是大于等于 0 的。

（4） 对 CM_o，CM_s，CM_r 使用**会计平衡证明**，证明交易满足会计平衡，即原始金额 = 交易金额 + 找零金额。

（5） 对 E(v_r)，E(v_r)' 分别使用**相等证明**，证明向接收者 R 发送的加密金额和向监管者 Q 发送的加密金额是相等的，即：E(v_r)，E(v_r)' 中的 v_r 是相同的。

上述所有零知识证明统称为Π。

最终用户 S 公布转账交易 tx = (SN_o， r_r1， CM_spk ，CM_rpk ，CM_o, CM_s, CM_r，E(v_r)，E(v_r)' ， Π)。

- **其中格式正确证明与验证过程如下：**

证明者证明密文的两部分：v * G1 +r1 * H 和 r2 * G2 中， r1 = r2。

证明者证明过程如下：

1) 证明者选择随机数 a, b。
2) 证明者计算 t1_p = a * G1 + b * H，t2_p = b * G2。
3) 证明者计算 c = Hash(t1_p, t2_p)
4) 证明者计算 z1 = a - c * v，z2 = b - c * r1

5) 证明者生成 format_proof =（c, z1, z2)。

验证者已知密文, G1, G2, H 和证明 format_proof，验证过程如下：

1) 验证者构造 t1_v = c * ciphertext_1 + z1 * G1 + z2 * H，t2_v = c * ciphertext_2 + z1 * G2。
2) 验证者检验 c = ? Hash(t1_v, t2_v)。若相等，则验证成功。

- **其中会计平衡证明与验证过程如下：**

证明者证明 CM_o = v_o * G1 + r_o1 ，CM_s = v_s * G1 + r_s1 * H，CM_r = v_r * G1 + r_r1 * H 中的金额 v_o、v_s、v_r 满足 v_r + v_s - v_o = 0。

证明者证明过程如下：

1) 证明者为交易金额的承诺 CM_r 选择随机数 a, b。
2) 证明者为找零金额的承诺 CM_s 选择随机数 d, e，为原始金额的承诺 CM_o 选择随机数 f。
3) 证明者计算 t1_p = a * G1 + b * H，t2_p = d * G1 + e * H，t3_p = (a + d) * G1 + f * H。
4) 证明者计算 c = Hash(t1_p, t2_p, t3_p)。
5) 证明者计算 R_v = a - c * v_r，R_r = b - c * r_r1。
6) 证明者计算 S_v = d - c * v_s,，S_r= e - c * r_s1，S_or = f - c * r_o1。
7) 证明者生成 balance_proof =（c, R_v, R_r, S_v, S_r, S_or)。

验证者已知 CM_s,CM_r, CM_o, G1, G2, H 和证明 balance_proof，验证过程如下：

1) 验证者计算

t1_v = R_v * G1 + R_r * H +c * CM_r,

t2_v = S_v * G1 + S_r * H +c * CM_s,

t3_v = (R_v + S_v) * G1 + S_or * H +c * CM_o。

2) 验证者检验 c = ? Hash(t1_v, t2_v, t3_v)。若相等，则验证成功。

5. 收款：

接收者 R 接收到转账交易 tx 后，使用自己的 Elgamal 私钥解密转账交易 tx 中的 E(v_r)，得到金额 v_r，接收者 R 验证该交易金额是否正确，若正确，则使用自己的地址私钥 R_sk 和转账交易 tx 中的 r_r1 计算该代币的序列号 SN_r，并存入自己钱包。

6. 验证：

大众看到公布的交易，可以验证：

（1） 通过检查 CM_o 是否属于承诺池，验证交易初始货币是否存在。

（2） 通过检查原始货币的序列号 SN_o 是否已经存在于区块链上，验证该货币之前是否已经被花费过。

(3) 通过验证交易中的零知识证明Π，验证交易内容是否合法有效。

7. 监管：

监管者看到公布的交易，用自己私钥解密其中的密文 E(v_r)’，查看本次转账交易的金额和交易双方的地址公钥。从而实现对交易金额和交易双方身份的监管。同时，验证零知识证明中的会计平衡证明和相等证明，确保本次转账交易的正确性与会计平衡。