Reported by YuTian Chen
CSU

Zerocash

Catalog

Introduction

ZEROCASH设计雏形

We construct a decentralized anonymous payment (DAP) scheme, which is a decentralized e-cash scheme that allows direct anonymous payments of any amount.

Here, we outline our construction.

在简化结构下，mint流程如下：

user - u

random serial number - sn

trapdoor - r

coin commitment - $cm := COMM_r(sn)$

$c := (r, sn, cm)$

对应的铸币交易tx_{mint}只包含cm (not including r & sn)

消费交易tx_{spend}包含

sn

zk-SNARK proof π of the NP statement:

"I know r such that $COMM_r(sn)$ appears in the list of coin commitments CMList."

(其中CMList由基于CRH的Merkle Tree数据结构存储)

user u生成密钥对(a_{pk}, a_{sk})，其中$a_{pk} := PRF_{a_{sk}}^{addr}(0)$.

重新设计后的mint流程：

① generate ρ (secret value)，$sn := PRF_{a_{sk}}^{sn}(\rho)$.

② $k := COMM_r(a_{pk}||\rho)$ for a random r.

$cm := COMM_s(v||k)$ for a random s, v is the mingting **value** of c.

coin $c := (a_{pk}, v, \rho, r, s, cm)$.

$tx_{mint} := (v, k, s, cm)$.

嵌套 commitment.

所以所有人能够验证tx_{mint}中的cm是否是一个价值v的铸币有效承诺.

$$if\ cm == COMM_s(v||k)$$

但无法获知$\underbrace{a_{pk}, sn(\text{从}\rho\text{中派生})}_{\text{hidden in k}}$

重新设计后的pour流程：

Consume the coin $c^{old} = (a_{pk}^{old}, v^{old}, \rho^{old}, r^{old}, s^{old}, cm^{old})$.

e.g. Produce two new coins = (c_1^{new}, c_2^{new}).

按Mint同样的流程得到$c_i^{new} = (a_{pk,i}^{new}, v_i^{new}, \rho_i^{new}, r_i^{new}, s_i^{new}, cm_i^{new})$.

于是u需要生成一个zk-SNARK证明$\pi_{pour}(for\ NP\ statements\ below)$.

$tx_{pour} := (rt, sn^{old}, cm_1^{new}, cm_2^{new}, \pi_{pour})$ *被加入账本.*

"Given rt (the Merkle-tree root), sn^{old}, cm_1^{new}, cm_2^{new}.
I knew c^{old}, c_1^{new}, c_2^{new}, a_{sk}^{old} (spent address secret key) such that:

- coin的形式均合法. 即 $k == COMM_r(a_{pk} \| \rho)$ and $cm == COMM_s(v \| k)$.
- $a_{pk}^{old} == PRF^{addr}_{a_{sk}^{old}}(0)$.
- $sn^{old} == PRF^{sn}_{a_{sk}^{old}}(\rho^{old})$.
- cm^{old} appears as a $\boxed{\text{leaf}}$ of a Merkle-tree with root rt. (Exited).
- $v_1^{new} + v_2^{new} == v^{old}$. "

u cannot **spend** new coins. &

u cannot **track** new user (u_1, u_2)

spending (c_1^{new}, c_2^{new}).

'cause he knows no info about revealed $sn_i^{new} := PRF^{sn}_{a_{sk,i}^{new}}(\rho_i^{new})$

Sending coins.

由于coin c中存在匿名字段，且避免使用带外(out-of-band)通道，所以作以下设计修改：
每个user持有密钥对$(addr_{pk}, addr_{sk})$.

(a_{pk}, pk_{enc})　　　(a_{sk}, sk_{enc})

user u计算$C_1 := E_{pk^{new}_{enc,1}}(v_1^{new}, \rho_1^{new}, r_1^{new}, s_1^{new})$，并将其包含在$tx_{pour}$中.

Receiver u_1可通过私钥$sk^{new}_{enc,1}$来解密C_1得到发送的coin.

u_2同理.

Public outputs.

上述的构造已满足mint/merge/split coins.
令$v_1^{new} + v_2^{new} + v_{pub} = v^{old}$ (增加v_{pub}字段，公开).
增设info字段用于记录redeem资产的dst (e.g. BTC pubkey，用于资产跨链回Bitcoin网络).
v_{pub}与info字段被包含入tx_{pour} (但public output为optional).

Non-malleability(Prevent malleability attack).

◎ sample a key pair(pk_{sig}, sk_{sig}) ← for a **one-time** signature.

◎ $h_{sig} := CRH(pk_{sig})$.

◎ $h_1 := PRF^{pk}_{a^{old}_{sk,1}}(h_{sig}), h_2 := PRF^{pk}_{a^{old}_{sk,2}}(h_{sig})$.

◎ 调整零知识证明相关NP statements.

◎ 用sk_{sig}签名得到σ (与pk_{sig}一起加入tx_{pour}).

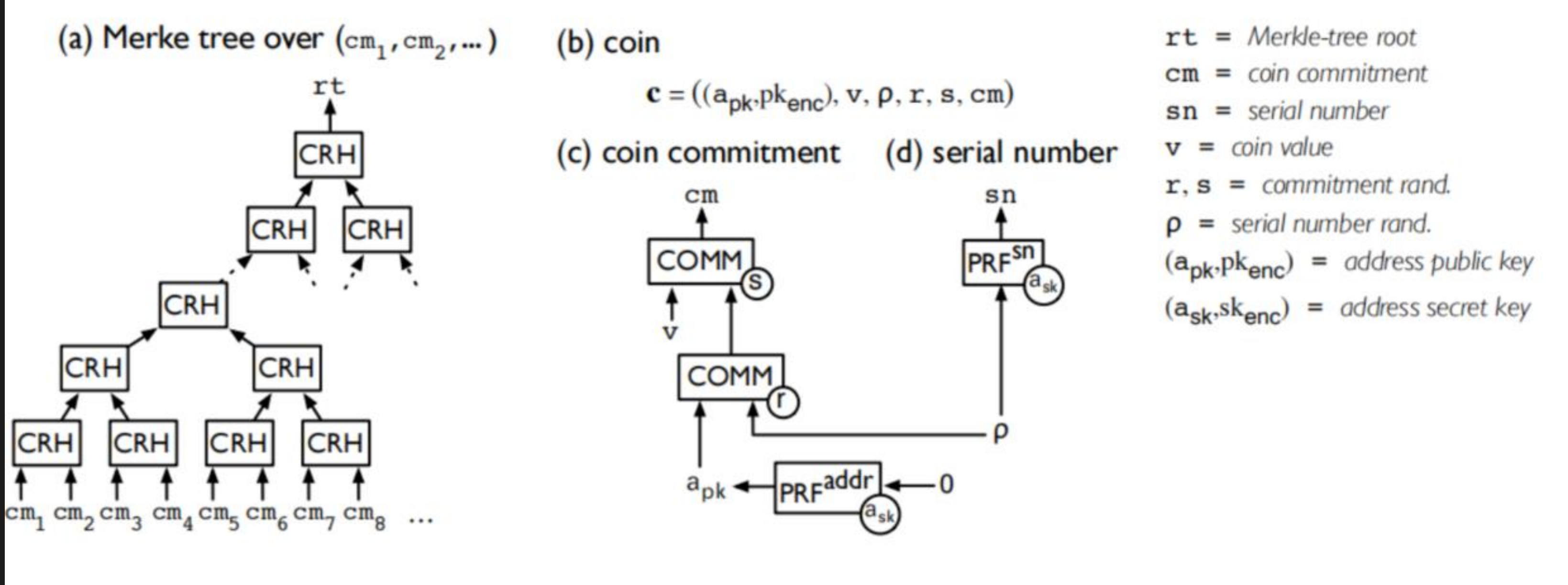


Figure 1: **(a)** Illustration of the CRH-based Merkle tree over the list CMList of coin commitments. **(b)** A coin **c**. **(c)** Illustration of the structure of a coin commitment cm. **(d)** Illustration of the structure of a coin serial number sn.

zk-SNARKs

ZERO-KNOWLEDGE

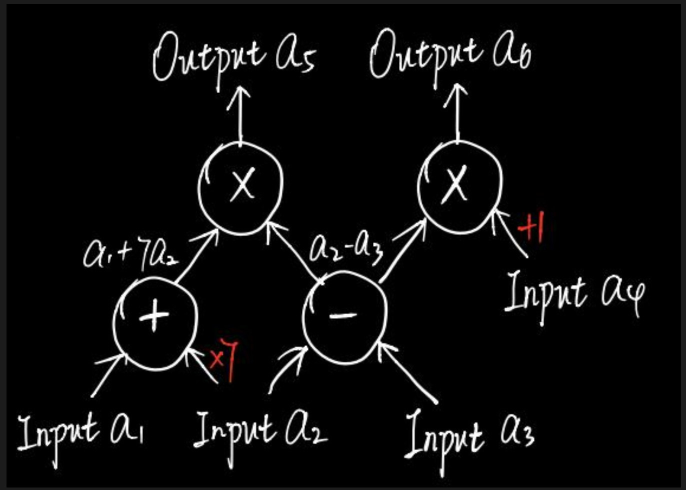
The purpose of this chapter is to present the construction of the proof π and the verification process in zk-SNARK.

Arithmetic Circuit.

由$C: F^n \times F^h \rightarrow F^l$. (算术电路，允许$\times$, $+$, $constant$，电路与门构成有向无环图)

e.g. $C: F_{11}^2 \times F_{11}^2 \rightarrow F_{11}^2$ is given by

$C(x_1, x_2, x_3, x_4) := ((x_1 + 7x_2)(x_2 - x_3), (x_2 - x_3)(x_4 + 1))$.



Quadratic Arithmetic Programs.

QAP的作用是为Prover构造proof π来证明其已知算术电路C的解.

$QAP := (\vec{A}, \vec{B}, \vec{C}, Z)$, 其中$\vec{A} = (A_i(z))_{i=0}^m, \vec{B} = (B_i(z))_{i=0}^m, \vec{C} = (C_i(z))_{i=0}^m, m \geq N = n + h + l$. (Z为target polynomial)

$$P(z) := \underbrace{(A_0(z) + \sum_{i=1}^m a_i A_i(z))}_{:=A(z)} \underbrace{(B_0(z) + \sum_{i=1}^m a_i B_i(z))}_{:=B(z)} - \underbrace{(C_0(z) + \sum_{i=1}^m a_i C_i(z))}_{:=C(z)}.$$

当且仅当$(a_1, ..., a_N)$为一组合法解时，

Prover将P(z)作为proof π，且Verifier可快速验证是否$\boldsymbol{Z(z) || P(z)}$.

Construction of QAPs.

考虑一个算术电路C，其电路输出都是乘法门的输出 (m=进入电路的所有输入线数+乘法门数).

◎ Preparation

令M为乘法门的集合，W为特殊线的集合 (即电路输入线+乘法门输出线).

for a gate g∈M，令$I_{g,L} \subset W$表示从left side进入g，且在此之前不经过其他乘法门的集合；$I_{g,R}$定义类似.

◎ Target Polynomial

$Z(z) = \prod_{g\in M}(z - r_g)\,,root\ r_i \in F.$

◎ Left and Right Input Polynomials

$A_i(r_g) = c_{g,L,i}\ if\ i \in I_{g,L},\ else\ A_i(r_g) = 0\,.$

$B_i(r_g) = c_{g,R,i}\ if\ i \in I_{g,R},\ else\ B_i(r_g) = 0\,.$

($c_{g,L,i}, c_{g,R,i}$为$i \in [1,m]$进入gate g的系数).

◎ Output Polynomial

$C_i(r_g) = 1\ if\ i = g,\ else\ 0.$

$$\therefore \begin{cases} A(r_g) = A_0(r_g) + \sum_{i=1}^{m} a_i A_i(r_g) = A_0(r_g) + \sum_{i\in I_{g,L}} a_i\, c_{g,L,i} \\ \text{(Input to g from left)} \\ B(r_g) = B_0(r_g) + \sum_{i=1}^{m} a_i B_i(r_g) = B_0(r_g) + \sum_{i\in I_{g,R}} a_i\, c_{g,R,i} \\ \text{(Input to g from right)} \\ C(r_g) = C_0(r_g) + \sum_{i=1}^{m} a_i C_i(r_g) = a_g \\ \text{(Output of g)} \end{cases}$$

$P(r_g) = A(r_g)B(r_g) - C(r_g) = 0.\ (for\ \forall g \in M).$

From QAP to zk-SNARKs.

define tow sets:

$R_C := \{(\vec{x}, \vec{w}) \in F^n \times F^h | C(\vec{x}, \vec{w}) = 0\}$ - “valid assignments with output of 0.”

$L_C := \{\vec{x} \in F^n | \exists \vec{w} \in F^h : C(\vec{x}, \vec{w}) = 0\}$ - “Language” (NP-complete), 其中$\vec{w}$为witness.

双线性对：假设G_1, G_2为两个阶为r的加法群，G_T为一个阶为r的乘法群.

Weil Pair, Tate Pair, ...

存在映射$e: G_1 \times G_2 \rightarrow G_T$.

满足$\boldsymbol{e(n_1 P_1, n_2 P_2) = e(P_1, P_2)^{n_1 n_2}}$，且$\boldsymbol{e(P_1, P_2) \neq 1_{G_T}}$ (非退化性).

编码方案：将$A_i(\tau), B_i(\tau)$映射到$A_i(\tau) P_1 \ in\ G_1, B_i(\tau) P_2 \ in\ G_2$.

$(P_1, P_2$为基点)

SNARK-Completeness.

① Key generation

1. Construct the QAP $Q(C) = (\vec{A}, \vec{B}, \vec{C}, Z)$ of C.

2. 随机选取$\tau, \rho_A, \rho_B \in F$，令$\rho_C = \rho_A \rho_B$.

3. **Proving Key**: $pk = (pk_A, pk_B, pk_C, pk_H)$.

$$pk_A := \underbrace{(A_i(\tau)\rho_A P_1)_{i=0}^{m}}_{pk_{A,i}} \ ; \ pk_B := \underbrace{(B_i(\tau)\rho_B P_2)_{i=0}^{m}}_{pk_{B,i}} \ ; \ pk_C := \underbrace{(C_i(\tau)\rho_C P_1)_{i=0}^{m}}_{pk_{C,i}} \ ; \ pk_H := \underbrace{(\tau^i P_1)_{i=0}^{d}}_{pk_{H,i}}$$

4. **Verifying Key**: $v_k = (vk_{IC}, vk_z)$.

$$vk_{IC} := \underbrace{(A_i(\tau)\rho_A P_1)_{i=0}^{n}}_{vk_{IC,i}} \ ; \ vk_z := Z(\tau)\rho_C P_2.$$

SNARK-Completeness.

② Prover

Prover已知一个合法$(\vec{x},\vec{w}) \in R_C$，需要找到所有乘法门的一个合法分布

Input: $pk, \vec{x} \in F^n, and\ \vec{w} \in F^h.$ (由多项式时间算法$QAP(C,\vec{x},\vec{w})$得到)

1. Construct the QAP $Q(C) = (\vec{A},\vec{B},\vec{C},Z)$ of C ;

2. Compute a valid distribution $(a_1,\ldots,a_m) = QAP(C,\vec{x},\vec{w})$;

3. $H(z) = \frac{A(z)B(z)-C(z)}{Z(z)}$，系数为$(h_i)_{i=0}^{d}$;

4. $proof\ \pi := (\pi_A, \pi_B, \pi_C, \pi_H)$

$$,\pi_A := \sum_{i=n+1}^{m} a_i pk_{A,i}\ ,\pi_B := pk_{B,0} + \sum_{i=1}^{m} a_i pk_{B,i}\ ,\pi_C := pk_{C,0} + \sum_{i=1}^{m} a_i pk_{C,i}\ ,\pi_H := pk_{H,0} + \sum_{i=1}^{d} h_i pk_{H,i}$$

Output: $proof\ \pi\ of\ \text{“}\vec{x} \in L_C\text{”}.$

5. Note that, π_H为H(z)在G_1上的编码，i.e. $\pi_H = H(\tau)P_1 \in G_1$.

$$\pi_B = (B_0(\tau) + \sum_{i=1}^{m} a_i B_i(\tau))\rho_B P_2 \in G_2.$$

…

SNARK-Completeness.

③ Verifier

$\boxed{\text{Input: } vk, \vec{x} \in F^n, proof\ \pi.}$

1. Calculate $vk_{\vec{x}} = vk_{IC,0} + \sum_{i=1}^{n} x_i vk_{IC,i}$;

2. Check QAP divisibility:

$$\boldsymbol{e(vk_{\vec{x}} + \pi_A, \pi_B) = e(\pi_H, vk_z) \cdot e(\pi_C, P_2)}$$

$$\Leftrightarrow\ \boldsymbol{e(P_1, P_2)^{\rho_A \rho_B A(\tau) B(\tau)} = e(P_1, P_2)^{\rho_C H(\tau) Z(\tau)} \cdot e(P_1, P_2)^{\rho_C C(\tau)}}$$

i.e. $P(\tau) = A(\tau)B(\tau) - C(\tau)$.

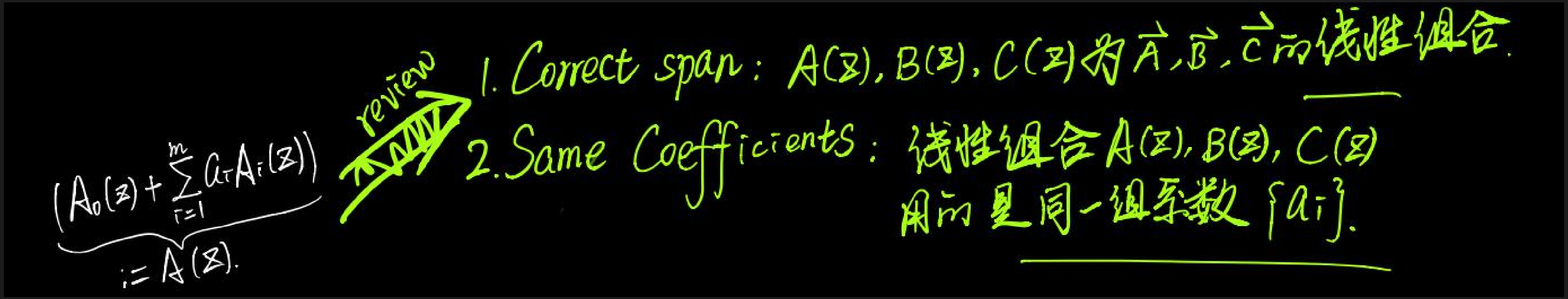
check whether $\boldsymbol{P(\tau) = H(\tau)Z(\tau)}$ or not.

SNARK-Soundness.

恶意Prover可构造$A(z) = 1, B(z) = Z(z), C(z) = 0$.

(尽管其不知道一组合法的$(a_i)_{i=0}^m$, i.e. $(\vec{x}, \vec{w}) \in R_C$, 但仍能通过Z(z)||P(z)的verify)

因为缺失以下check：



所以我们作以下修改/增加：

$let\ pk'_{A,i} = \alpha_A pk_{A,i}$.

Prover does not know α_A, but manages to contruct $\pi'_A = \alpha_A \pi_A$.

Key generation Extras

check for (1).

1. Randomly sample $\alpha_A, \alpha_B, \alpha_C \in F$.
2. Proving key:

$$pk_A' = \left(\alpha_A A_i(\tau)\rho_A P_1\right)_{i=0}^{m}.$$

$$pk_B' = \left(\alpha_B B_i(\tau)\rho_B P_1\right)_{i=0}^{m}$$

$$pk_C' = \left(\alpha_C C_i(\tau)\rho_C P_1\right)_{i=0}^{m}.$$

3. Verifying key:

$$vk_A = \alpha_A P_2. \quad vk_B = \alpha_B P_1. \quad vk_C = \alpha_C P_2.$$

check for (2).

1. Randomly sample $\beta, \gamma \in F$.
2. Proving key:

$$pk_K = \underbrace{\left(\beta(\rho_A A_i(\tau) + \rho_B B_i(\tau) + \rho_C C_i(\tau))P_1\right)}_{pk_{K,i}}{}_{i=0}^{m}.$$

3. Verifying key:

$$vk_\gamma = \gamma P_2, \quad vk_{\beta\gamma}^1 = \gamma\beta P_1,$$

$$vk_{\beta\gamma}^2 = \gamma\beta P_2.$$

Ensure correct span.

- For Prover:

$$\pi_A' = \sum_{i=n+1}^{m} a_i pk'_{A,i}. \qquad \pi_B' = pk'_{B,0} + \sum_{i=1}^{m} a_i pk'_{B,i}. \qquad \pi_C' = pk'_{C,0} + \sum_{i=1}^{m} a_i pk'_{C,i}$$

- For Verifier: check

$$e(\pi_A, vk_A) = e(\pi_A', P_2). \qquad e(vk_B, \pi_B) = e(\pi_B', P_2). \qquad e(\pi_C, vk_C) = e(\pi_C', P_2).$$

当且仅当 $\pi_A' = \alpha_A \pi_A$ 时等式成立，可认为 prover 使用了正确的 $pk_{A,i}$ & $pk'_{A,i}$ of $\vec{A}$.

Ensure same coefficients.

For Prover:

$$\pi_K = pk_{K,0} + \sum_{i=1}^{m} a_i pk_{K,i}$$

For Verifier:

$$e(\pi_K, vk_\gamma) = e(vk_{\vec{x}} + \pi_A + \pi_C, vk^2_{\beta\gamma}) \cdot e(vk^1_{\beta\gamma}, \pi_B)$$

$$\Updownarrow$$

$$e(P_1,P_2)^{\beta(\rho_A A(\tau)+\rho_B B(\tau)+\rho_C C(\tau))\cdot\gamma} = e(P_1,P_2)^{(\rho_A A(\tau)+\rho_C C(\tau))\gamma\beta} \cdot e(P_1,P_2)^{\gamma\beta\cdot\rho_B B(\tau)}$$

'cause β is hidden & use $(\rho_A A(\tau)+\rho_B B(\tau)+\rho_C C(\tau))$.

当校验通过后，可认为使用了同一组解 $\{a_i\}$s. ②

Formal

CONSTRUCTION OF ZEROCASH

In this chapter, we provide a formal construction of zerocash and its optimizations.

Cryptographic building blocks.

(λ denotes the security parameter)

CRH: $\{0,1\}^* \rightarrow \{0,1\}^{O(\lambda)}$;

$PRF_x^{addr}(z) = PRF_x(00||z)$, $PRF_x^{sn}(z) = PRF_x(01||z)$ $PRF_x^{pk}(z) = PRF_x(10||z)$;

(PRF is also collision resistant)

zk-SNARKs for pouring coins.

Recall: (When a user u pours “old” coins c_1^{old}, c_2^{old} into new coins c_1^{new}, c_2^{new})

$tx_{pour} = (rt, sn_1^{old}, sn_2^{old}, cm_1^{new}, cm_2^{new}, v_{pub}, info, *)$

我们需要在”*”中提供证明满足pour operation的各种条件.

NP Statement POUR is defined as follows:

1. Instances are of the form $\boldsymbol{x = (rt, sn_1^{old}, sn_2^{old}, cm_1^{new}, cm_2^{new}, v_{pub}, h_{sig}, h_1, h_2)}$

2. Witnesses are of the form $\boldsymbol{a = (path_1, path_2, c_1^{old}, c_2^{old}, addr_{sk,1}^{old}, addr_{sk,2}^{old}, c_1^{new}, c_2^{new})}$

$$
\begin{cases}
c_i^{old} = (addr_{pk,i}^{old}, v_i^{old}, \rho_i^{old}, r_i^{old}, s_i^{old}, cm_i^{old}) \\
c_i^{bew} = (addr_{pk,i}^{new}, v_i^{new}, \rho_i^{new}, r_i^{new}, s_i^{new}, cm_i^{new}) \\
addr_{pk,i}^{old} = (a_{pk,i}^{old}, pk_{enc,i}^{old}) \\
addr_{pk,i}^{new} = (a_{pk,i}^{new}, pk_{enc,i}^{new}) \\
addr_{sk,i}^{old} = (a_{sk,i}^{old}, sk_{enc,i}^{old})
\end{cases}
$$

TODO: paper中未提及check使用$pk_{enc,i}^{new}$加密coin的正确性
(倘若恶意send fake coin， Receiver无法解密出正确的ρ，进而无法得到sn)

a witness **a** is valid for **x** if the following holds:

(a) cm_i^{old} of c_i^{old} appears on the ledge (CRH-based Merkle Tree of root rt).

i.e. $path_i$ is valid.

(b) $a_{pk,i}^{old} = PRF^{addr}_{a^{old}_{sk,i}}(0)$.

(c) $sn_i^{old} = PRF^{sn}_{a^{old}_{sk,i}}(\rho_i^{old})$.

$$
\left.
\begin{aligned}
&(d)\ cm_i^{old} = COMM_{s_i^{old}}(COMM_{r_i^{old}}(a_{pk,i}^{old} \| \rho_i^{old}) \| v_i^{old}). \\
&(e)\ cm_i^{new} = COMM_{s_i^{new}}(COMM_{r_i^{new}}(a_{pk,i}^{new} \| \rho_i^{new}) \| v_i^{new}).
\end{aligned}
\right\} \text{coin is well-formed.}
$$

(f) $h_i = PRF^{pk}_{a^{old}_{sk,i}}(i \| h_{sig})$.

(g) $v_1^{new} + v_2^{new} + v_{pub} = v_1^{old} + v_2^{old}$. (with $v_1^{old}, v_2^{old} \geq 0$, $v_1^{old} + v_2^{old} \leq v_{max}$).

Zerocash.

Operation tuple Π = (Setup, CreateAddress, Mint, **Pour**, VerifyTransaction, Receive).

(见paper/Note)

Instantiation of building blocks.

>>>

Let $\mathcal{H}$ be the SHA256 compression function (CRH)

(512 bytes to 256 bytes, 满足构建Merkle Tree(two-to-one)设计)

$\boldsymbol{PRF_x(z)} = \mathcal{H}(x||z)$ *with* $x \in \{0,1\}^{256}$ *as the seed, and* $z \in \{0,1\}^{256}$ *as the input.*

>>>

$k = COMM_r(a_{pk}||\rho) = \mathcal{H}(r||[\mathcal{H}(a_{pk}||\rho)]_{128})$;

$cm = COMM_s(v||k)$;

>>>

Signature: using ECDSA

(且由于standard ECDSA中(r, s), (r, -s)均可验签，因此与Bitcoin作同样s截取”lower half”的处理)

>>>

TODO

Something unread.

Arithmetic circuit for Operations in Zerocash.

paper中verify SHA256的电路生成采用手工优化,
相较于一般性的(如libsnark)第三方库生成的电路门有显著减少.

TODO => libsnark源码阅读
https://mp.weixin.qq.com/s?__biz=MzU5MzMxNTk2Nw==&mid=2247486482&idx=1&sn=407d59e7fc47de0e929c653ce00eb260&chksm=fe131d02c9649414b27a0684ce950b2a63a84ca9c901f81b7251964befd3388e0f616df5bd4b&scene=21#wechat_redirect

还涉及不同描述语言之间的转换(e.g. r1cs -> qap)

Thx