참여자

Server : CEX 서비스 제공자(중앙 거래소), 총자산을 증명하려는 주체

Smart Contract : 거래소의 총자산과 커밋이 저장되어 있어 누구나 확인 할 수 있다.

Users : CEX 서비스 사용자

Notation

$AccountList$(AL): list of cc-SNARK proofs that contain Pedersen commitment of value $(\pi_{cc_0}, \cdots, \pi_{cc_n})$. Size of list can be incremental. All users of CEX can view account LIst

$cmInfo$: private database of Pedersen Commitment opening keys $\{(r_0, v_0)\cdots, (r_n, v_n)\}$

$\pi_{cc}.d$: cc-SNARK proof d term contain pedersen commitment of account value where $\pi_{cc}.d = g^v h_r$

C_{cex} : total value commitment. Using Pedersen Commitment's homomorphic property, server can product total account commitment to get the total value commitment. $C_{cex} = g^v h^{r_{cex}}$

cc-SNARK:

setup(R) returns {pk, vk, pp(G,p,g,h)}

prove(pk, $\vec{x}$, $\vec{w}$) returns {π, c_w, r} : c_w는 $\vec{w}$의 vector pedersen commitment로 proof의 D term ($c_w = \prod g_i^{w_i} \cdot h^r$)

Verify(vk, π, $\vec{x}$) returns 1/0 cc-SNARK

$\mathsf{LinSetup}(R_Q) \to (\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$

$\alpha, \beta, \gamma, \delta, \eta, \tau \leftarrow\$\, \mathbb{F}^*$

$$\boldsymbol{\sigma}_1 := \left(1, \alpha, \beta, \delta, \{\tau^i\}_{i=1}^{d-1}, \boxed{\left\{\frac{\beta a_i(\tau) + \alpha b_i(\tau) + c_i(\tau)}{\gamma}\right\}_{i=1}^{n}, \frac{\eta}{\gamma}} \left\{\frac{\beta a_i(\tau) + \alpha b_i(\tau) + c_i(\tau)}{\delta}\right\}_{i=n+1}^{m}, \left\{\frac{1}{\delta}\tau^i t(\tau)\right\}_{i=0}^{d-2}, \frac{\eta}{\delta}\right)$$

$$\boldsymbol{\sigma}_2 := \left(1, \beta, \gamma, \delta, \{\tau^i\}_{i=0}^{d-1}\right)$$

$\mathsf{ProofMatrix}(R_Q, \boldsymbol{w}) \to (\Pi_1, \Pi_2)$

Let $\boldsymbol{w} := (\boldsymbol{u}, \boldsymbol{\omega})$. Compute $h(Z)$ as in (9) ; $r, s, v \leftarrow\$\, \mathbb{F}$

Let $\Pi_1 \in \mathbb{F}^{3\times(m+2d+6)}, \Pi_2 \in \mathbb{F}^{1\times(d+4)}$ s.t. $(A, C, D)^\top = \Pi_1 \cdot \boldsymbol{\sigma}_1, B = \Pi_2 \cdot \boldsymbol{\sigma}_2$ and

$$A := \alpha + \sum_{k=0}^{m} w_k \cdot a_k(\tau) + r\delta \quad ; \quad B := \beta + \sum_{k=0}^{m} w_k \cdot b_k(\tau) + s\delta \quad ; \quad \boxed{D := \sum_{k=0}^{n} w_k \cdot \frac{1}{\gamma}(\beta a_k(\tau) + \alpha b_k(\tau) + c_k(\tau)) + v\frac{\eta}{\gamma}}$$

$$C := \sum_{k=n+1}^{m} w_k \cdot \frac{\beta a_k(\tau) + \alpha b_k(\tau) + c_k(\tau)}{\delta} - v\frac{\eta}{\delta} + \sum_{i=0}^{d-2} h_i \frac{\tau^i t(\tau)}{\delta} + As + Br - rs\delta$$

$\mathsf{Test}(R_Q) \to T$

Define $T \in \mathbb{F}^{(m+2d+9)\times(d+5)}$ encoding the following quadratic test: $A \cdot B = \alpha \cdot \beta + C \cdot \delta + D \cdot \gamma$

Figure 22: Our NILP for an augmented QAP relation $R_Q(\boldsymbol{u}, \boldsymbol{\omega})$, to be used to obtain ccGro16.

$$\mathsf{VerCommit}(\mathsf{ck}, [D_1], \boldsymbol{u}, o) := [D]_1 \stackrel{?}{=} \sum_{k=0}^{n} u_k \cdot \left[\frac{1}{\gamma}(\beta a_k(\tau) + \alpha b_k(\tau) + c_k(\tau))\right]_1 + o \cdot \left[\frac{\eta}{\gamma}\right]_1$$

- 계좌 잔고의 value가 양수임을 증명하는데 cc-SNARK 사용
- Proof안에 witness의 commitment가 포함되어있음. Circuit 내부에서 commitment 체크할 필요가 없어진다.

Account value range proof cc-SNARK Circuit Relation(;v):

Check $0 \leq v < 2^{64}$

- 스마트컨트랙트에 업데이트하기 위해서 C_{cex}를 잘 만들었는지 확인
- 업데이트를 잘했는지도 필요할까 ? 밑에 두개 중 하나 선택

1. CEX total value update SNARK Circuit Relation($C_{cex}, v_{cex}, C_{cex}', v_{cex}'; \ r_{cex}, r_{cex}', c_i, c_i'$):

$$c_{cex} = g^{v_{cex}} h^{r_{cex}}$$
$$c_{cex}' = g^{v_{cex}'} h^{r_{cex}'}$$
$$c_{cex}' = c_{cex} \cdot \ c_i^{-1} \cdot \ c_i'$$

2. CEX total value update SNARK Circuit Relation($C_{cex}, v_{cex}, ; \ r_{cex}$):

$$c_{cex} = g^{v_{cex}} h^{r_{cex}}$$

Server Algorithm : setup, newAccount, updateAccount

AccountList(AL) : $\{\pi_{cc_0}, \cdots, \pi_{cc_N}\}$ list of cc-snark proof

cmInfo : { $(r_0, v_0), \cdots, (r_n, v_n)$ } list of opening key of cm

<u>setup(R_{cc}, R) :</u>

$pk_{cc}, vk_{cc}, pp_{ped} \leftarrow \Pi_{cc}.Setup(R_{cc})$

$pk, vk \leftarrow \Pi.setup(R)$

$C_{cex} \leftarrow 1$

Returns $(pk, vk, pk_{cc}, vk_{cc}, c_{cex})$

<u>newAccount(pk_{cc}, pk i : index of newAccount) :</u>

$\pi_{cc_i}, c_i, r_i \leftarrow \Pi_{cc}.Prove(pk_{cc}, ; 0)$

$C_{cex} \leftarrow C_{cex} \times c_i$ where $C_{cex} = pp_{ped}.g^v pp_{ped}.h^r$

$r_{cex} \leftarrow r_{cex} + r_i$

$\pi_{cex} \leftarrow \Pi.Prove(pk, C_{cex}, v_{cex}; r_{cex})$

AL.append(π_{cc_i})

cmInfo.append((0, r_i))

Return $Tx_{update} = (\pi_{cex},\ C_{cex},\ v_{cex}), r_i$

<u>updateAccount($pk_{cc},\ pk,$ i : index of newAccount, v_i') :</u>

$\pi_{cc_i}', c_i', r_i' \leftarrow \Pi_{cc}.Prove(pk_{cc}, v_i')$

Get c_i from AL and (v_i, r_j) from cmInfo

$C_{cex} \leftarrow C_{cex} c_i' c_i^{-1}$

$r_{cex} \leftarrow r_{cex} + r_i' - r_i$

$v_{cex} \leftarrow v_{cex} + v_i' - v_i$

$\pi_{cex} \leftarrow \Pi.Prove(pk, C_{cex}, v_{cex}; r_{cex})$

AL[i] $\leftarrow \pi_{cc_i}$

cmInfo[i] $\leftarrow (r_i', v_i')$

Return $Tx_{update} = (\pi_{cex},\ C_{cex},\ v_{cex}), r_i'$

SmartContract all users can view C_{cex}, v_{cex} to check CEX server has an asset more than v_{cex} and to make sure C_{cex} is well made.

<u>setup(vk) :</u>

Store zk-snark verification key vk

<u>update(π_{cex}, , C_{cex}, v_{cex}) :</u>

Assert msg.sender == contract Owner

Assert $\Pi . Verify(vk, \pi_{cex}, C_{cex}, v_{cex})$

Store C_{cex}, v_{cex} as public

newAccount(pk_{cc}, pk i : index of newAccount) :

$\pi_{cc_i}, c_i, r_i \leftarrow \Pi_{cc}.Prove(pk_{cc}, ;0)$

$C_{cex} \leftarrow C_{cex} \times c_i$ where $C_{cex} = pp_{ped}.g^v pp_{ped}.h^r$

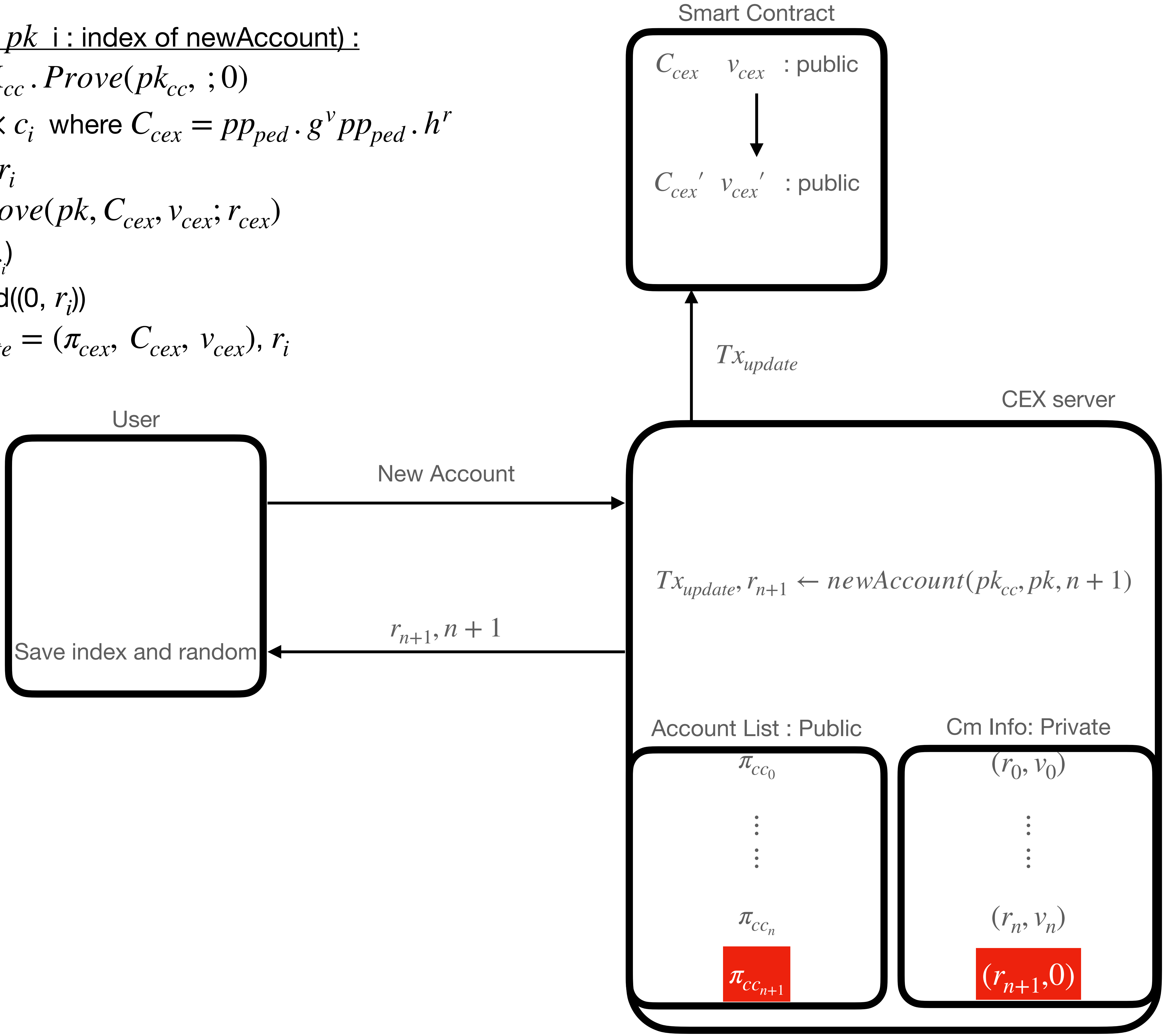
$r_{cex} \leftarrow r_{cex} + r_i$

$\pi_{cex} \leftarrow \Pi.Prove(pk, C_{cex}, v_{cex}; r_{cex})$

AL.append(π_{cc_i})

cmInfo.append((0, r_i))

Return $Tx_{update} = (\pi_{cex}, C_{cex}, v_{cex}), r_i$



updateAccount(pk_{cc}, pk, i : index of newAccount, v_i') :

$\pi_{cc_i}', c_i', r_i' \leftarrow \Pi_{cc}.Prove(pk_{cc}, v_i')$

Get c_i from AL and (v_i, r_j) from cmInfo

$C_{cex} \leftarrow C_{cex} c_i' c_i^{-1}$

$r_{cex} \leftarrow r_{cex} + r_i' - r_i$

$v_{cex} \leftarrow v_{cex} + v_i' - v_i$

$\pi_{cex} \leftarrow \Pi.Prove(pk, C_{cex}, v_{cex}; r_{cex})$

AL[i] $\leftarrow \pi_{cc_i}$

cmInfo[i] $\leftarrow (r_i', v_i')$

Return $Tx_{update} = (\pi_{cex}, C_{cex}, v_{cex}), r_i'$