Enhancing Permissioned Blockchains with Controlled Data Authorization

Qichang Liu, Xufeng Zhang, Sisi Duan, Haibing Zhang

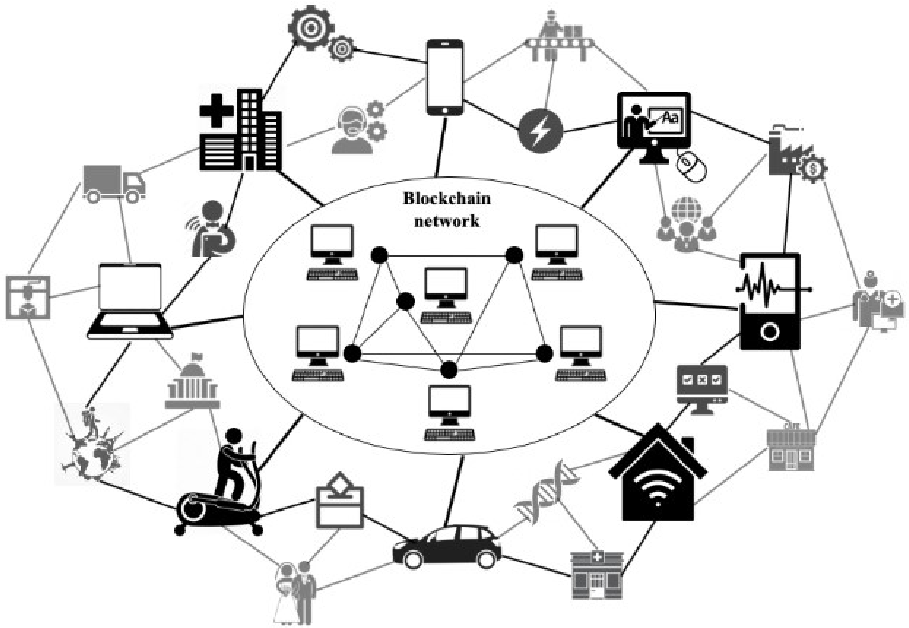
July 17 2024

清華大學 Tsinghua University

Contents

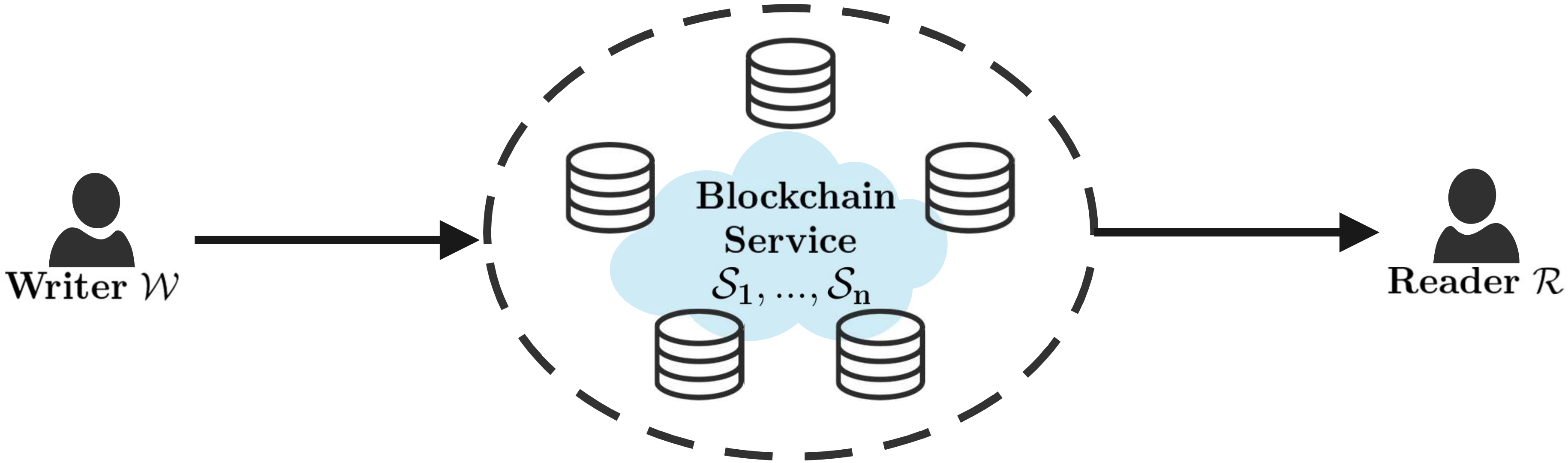
Tsinghua University

Goals of permissioned blockchains



- Data consistency √
- Data availability √
- Data confidentiality ?

Confidentiality goals in permissioned blockchains



- Causality preservation
- Confidentiality with access control

Threshold encryption with controlled authorization (ECA)

Contribution

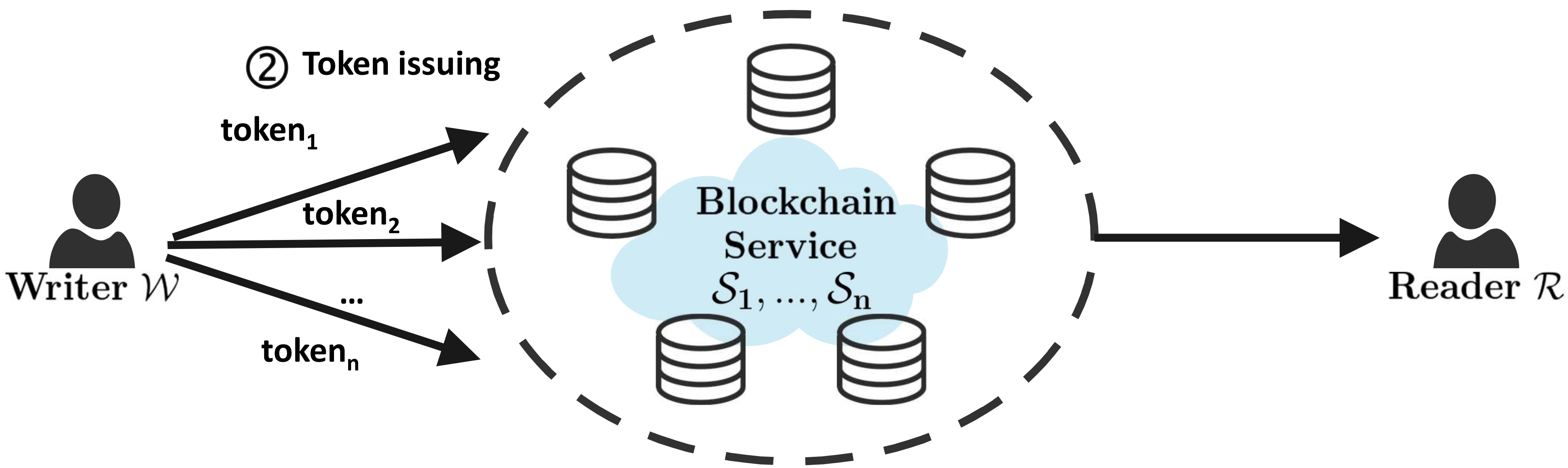
- Defined the primitive of threshold encryption with controlled authorization (ECA), along with three security notions.
- Provided a novel ECA scheme satisfying the security notions we define.
- Developed formal security proofs for our ECA scheme with a novel security reduction to the square Decisional Diffie-Hellman (DDH) assumption.

Contents

ECA: Authorization

Key generation
$(pk_W, sk_W) \leftarrow$ KeyGen1

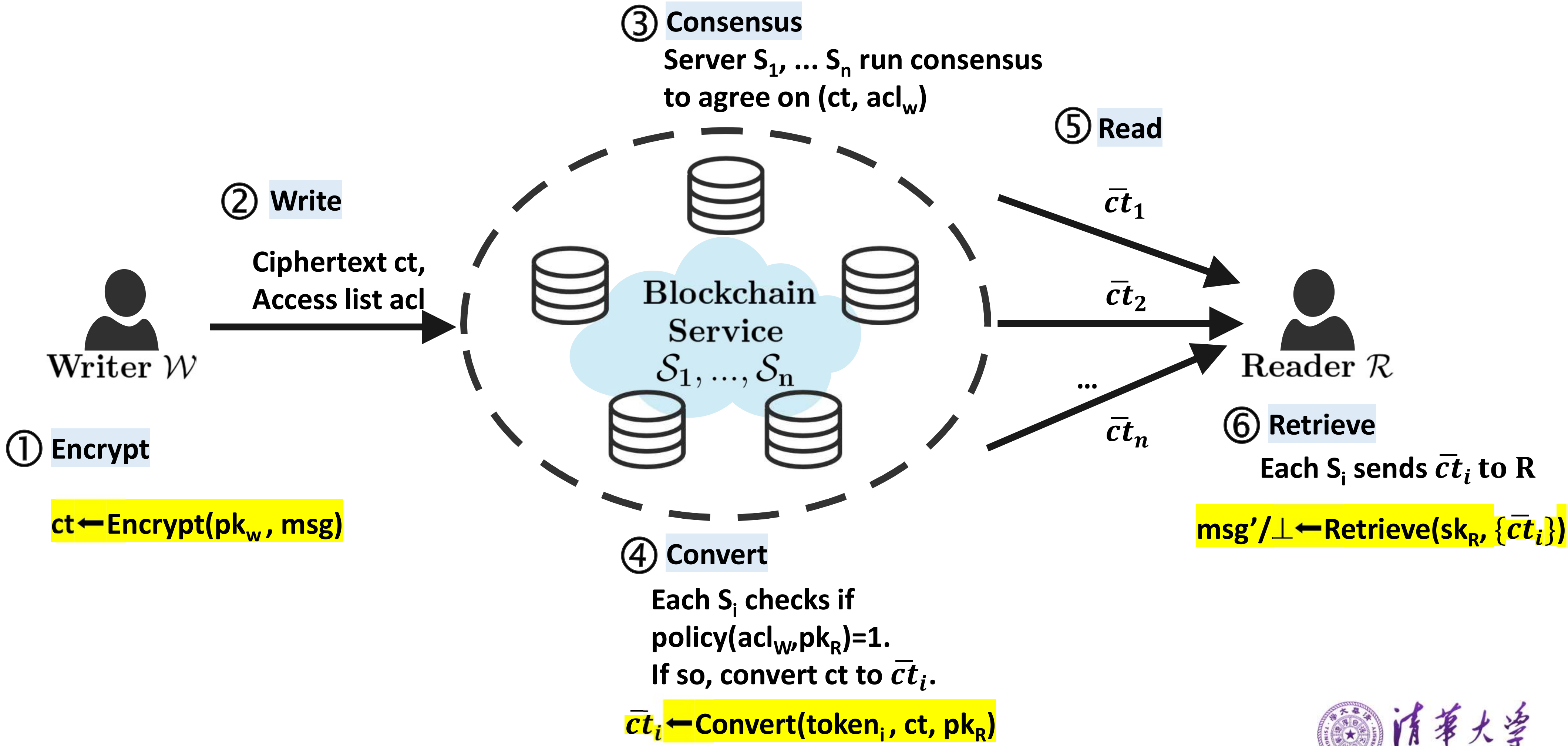
Key generation
$(pk_R, sk_R) \leftarrow$ KeyGen2



① **Token authorization**
$(token_1, \ldots, token_n) \leftarrow$ Authorize

③ **Token certification**
$0/1 \leftarrow$ Certify($token_i$)

ECA: Encrypted data transmission



ECA: Threshold Encryption with Controlled Authorization

ECA = (KeyGen1, KeyGen2, Authorize, Certify, Encrypt, Convert, Retrieve)

- **$(pk_W, sk_W) \leftarrow$ KeyGen1**
- **$(pk_R, sk_R) \leftarrow$ KeyGen2**
- **$(token_1, \ldots, token_n) \leftarrow$ Authorize(sk_W)**
- **0/1 $\leftarrow$ Certify($token_i$)**
- **ct $\leftarrow$ Encrypt(pk_W, msg)**
- **$\overline{ct}_i \leftarrow$ Convert($token_i$, ct, pk_R)**
- **msg'/$\perp \leftarrow$ Retrieve(sk_R, $\{\overline{ct}_i\}$)**

- **Token verifiability**
  $(token_1, \ldots, token_n) \leftarrow$ Authorize(sk_W)
  1 $\leftarrow$ Certify($token_i$)
- **t-robustness**
  ct $\leftarrow$ Encrypt(pk_W, msg)
  $\overline{ct}_i \leftarrow$ Convert($token_i$, ct, pk_R)
  Retrieve(sk_R, $\{\overline{ct}_i\}$)=msg
  If at most t fake $\overline{ct}_i$s
- **t-cr-CPA security**
  ct $\leftarrow$ Encrypt(pk_W, msg)
  The computational indistinguishability of ct and $\overline{ct}_i$ for any two plaintexts of the adversary's choice, even if the adversary corrupts at most t servers by obtaining their tokens

Tsinghua University

Contents

Tsinghua University

How to achieve ECA?

- (t, n) - Verifiable Secret Sharing

$(pk_W, sk_W) \leftarrow$ KeyGen1

$(token_1, \ldots, token_n) \leftarrow$ Authorize(sk_W)

$0/1 \leftarrow$ Certify$(token_i)$

Token verifiability

- (t,n) – Threshold Proxy Re-Encryption(CPA-security)

$(pk_W, sk_W) \leftarrow$ KeyGen1

$ct \leftarrow$ Encrypt(pk_w, msg)

$(pk_R, sk_R) \leftarrow$ KeyGen2

$\bar{ct}_i \leftarrow$ Convert$(token_i, ct, pk_R)$

Retrieve$(sk_R, \{\bar{ct}_i\})$=msg

t-cr-CPA-security

t-robustness

- Non-Interactive Zero Knowledge Proof

Instantiation of the ECA system

- **(t, n) - Verifiable Secret Sharing over a bilinear group**

$(pk_{\mathcal{W}}, sk_{\mathcal{W}}) \leftarrow \mathsf{KeyGen1}(pp)$:
$\alpha \leftarrow_\$ \mathbb{Z}_p$, $v_0 := g_T^{\alpha}$.
For $j \in [t]$: $r_j \leftarrow_\$ \mathbb{Z}_p$, $v_j := g_T^{r_j}$.
$sk_{\mathcal{W}} := (\alpha, r_1, \ldots, r_t)$.
$r(x) := \alpha + \sum_{j=1}^{t} r_j \cdot x^j \mod p$.
For $i \in [n]$:
  $\theta_i := r(i)$, $w_i = e(g_1^{1/\alpha}, g_2^{\theta_i})$.
$pk_{\mathcal{W}} := (g_1^{1/\alpha}, (v_0, v_1, \ldots, v_t), (w_1, \ldots, w_n))$.
Return $(pk_{\mathcal{W}}, sk_{\mathcal{W}})$.

Verifiable Property

$(\mathsf{token}_1, \ldots, \mathsf{token}_n) \leftarrow \mathsf{Authorize}(sk_{\mathcal{W}})$:
Parse $sk_{\mathcal{W}} = (\alpha, r_1, \ldots, r_t)$.
$r(x) := \alpha + \sum_{j=1}^{t} r_j \cdot x^j \mod p$.
For $i \in [n]$:
  $\theta_i := r(i)$.
  $\mathsf{token}_i := \theta_i$.
Return $(\mathsf{token}_1, \ldots, \mathsf{token}_n)$.

(t,n)-Secret Sharing

$0/1 \leftarrow \mathsf{Certify}(pk_{\mathcal{W}}, i, \mathsf{token}_i)$: // validity of tokens
Parse $pk_{\mathcal{W}} = (v', (v_0, v_1, \ldots, v_t), (w_1, w_2, \ldots, w_n))$.
Parse $\mathsf{token}_i = \theta_i$.
//test consistency between θ_i, w_i and $(v', v_0, \ldots, v_t)$
If $\left(g_T^{\theta_i} \neq \prod_{\ell=0}^{t} v_\ell^{i^\ell}\right) \vee \left(e(v', g_2^{\theta_i}) \neq w_i\right)$:
  Return 0. //token invalid
For $k \in [t+1]$, $j \in [t+2, n]$:
  $\lambda_{kj} := \prod_{i=1, i \neq k}^{n} \frac{j-i}{k-i}$. //Lagrange coefficients
//test consistency between v' and $(w_1, \ldots, w_{t+1})$
If $(\prod_{k=1}^{t+1} w_k^{\lambda_{k0}} \neq g_T)$: Return 0; //token invalid
//test consistency of $(w_1, \ldots, w_n)$
For $j \in [t+2, n]$
  If $\prod_{k=1}^{t+1} w_k^{\lambda_{kj}} \neq w_j$: Return 0; //token invalid
Else: Return 1. //token valid

Verifiable Property

Instantiation of the ECA system

- **(t,n)-Threshold Proxy Re-Encryption**

$\underline{(\mathsf{ct}, \mathsf{acl}_{\mathcal{W}}) \leftarrow \mathsf{Encrypt}(pk_{\mathcal{W}}, \mathsf{msg}, \mathsf{acl}_{\mathcal{W}}):}$

Parse $pk_{\mathcal{W}} = (v', (v_0, v_1, \ldots, v_t), (w_1, w_2, \ldots, w_n))$.
$r \leftarrow_{\$} \mathbb{Z}_p$, $\mathsf{ct}_1 := v'^r$, $k := g_T^r$.
$\mathsf{ct}_2 \leftarrow \mathsf{SE.Enc}(k, \mathsf{msg})$.
Return $(\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2), \mathsf{acl}_{\mathcal{W}})$.

$\underline{(pk_{\mathcal{R}}, sk_{\mathcal{R}}) \leftarrow \mathsf{KeyGen2}(\mathsf{pp}):}$

$\beta \leftarrow_{\$} \mathbb{Z}_p$, $sk_{\mathcal{R}} := \beta$, $pk_{\mathcal{R}} := g_2^{\beta}$.
Return $(pk_{\mathcal{R}}, sk_{\mathcal{R}})$.

- **Non-Interactive Zero Knowledge Proof**

$\underline{\bar{\mathsf{ct}}_i \leftarrow \mathsf{Convert}(\mathsf{token}_i, (pk_{\mathcal{W}}, \mathsf{ct}, \mathsf{acl}_{\mathcal{W}}), pk_{\mathcal{R}} = g_2^{\beta}, \mathsf{policy}):}$

If $\mathsf{policy}(\mathsf{acl}_{\mathcal{W}}, pk_{\mathcal{R}}) = 0$: Return $\perp$.
Parse $\mathsf{token}_i = \theta_i$.
Parse $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$.
$h := e(\mathsf{ct}_1, pk_{\mathcal{R}}) = e(\mathsf{ct}_1, g_2^{\beta})$, $\bar{h}_i := h^{\theta_i}$.
$//\pi_i \leftarrow \mathsf{ZKProof}((h, \bar{h}_i, g_T, \bar{u}_i), \theta_i)$
$\bar{u}_i := g_T^{\theta_i}$.
$s_i \leftarrow_{\$} \mathbb{Z}_p$, $\hat{h}_i := h^{s_i}$, $\hat{u}_i := g_T^{s_i}$.
$e_i := H(h, g_T, \bar{h}_i, \bar{u}_i, \hat{h}_i, \hat{u}_i)$.
$f_i := s_i + \theta_i \cdot e_i \mod p$.
$\pi_i := (\hat{h}_i, \hat{u}_i, f_i)$. $\quad$ //proof of $\log_h \bar{h}_i = \log_{g_T} \bar{u}_i$
$\bar{\mathsf{ct}}_{i,0} := h$.
$\bar{\mathsf{ct}}_{i,1} := (\bar{h}_i, \pi_i)$, $\bar{\mathsf{ct}}_{i,2} := \mathsf{ct}_2$.
Return $\bar{\mathsf{ct}}_i = (\bar{\mathsf{ct}}_{i,0}, \bar{\mathsf{ct}}_{i,1}, \bar{\mathsf{ct}}_{i,2})$.

Proof Generation: Prove that $\bar{ct}_i$ is generated by the correct token θ_i

Instantiation of the ECA system

- **(t,n)-Threshold Proxy Re-Encryption**

$\bar{\mathsf{ct}}_i \leftarrow \mathsf{Convert}(\mathsf{token}_i, (pk_{\mathcal{W}}, \mathsf{ct}, \mathsf{acl}_{\mathcal{W}}), pk_{\mathcal{R}} = g_2^{\beta}, \mathsf{policy})$:

If $\mathsf{policy}(\mathsf{acl}_{\mathcal{W}}, pk_{\mathcal{R}}) = 0$: Return $\perp$.
Parse $\mathsf{token}_i = \theta_i$.
Parse $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$.
$h := e(\mathsf{ct}_1, pk_{\mathcal{R}}) = e(\mathsf{ct}_1, g_2^{\beta}), \quad \bar{h}_i := h^{\theta_i}$.
$//\pi_i \leftarrow \mathsf{ZKProof}((h, \bar{h}_i, g_T, \bar{u}_i), \theta_i)$

$\bar{u}_i := g_T^{\theta_i}$.
$s_i \leftarrow_{\$} \mathbb{Z}_p, \quad \hat{h}_i := h^{s_i}, \quad \hat{u}_i := g_T^{s_i}$.
$e_i := H(h, g_T, \bar{h}_i, \bar{u}_i, \hat{h}_i, \hat{u}_i)$.
$f_i := s_i + \theta_i \cdot e_i \mod p$.

$\pi_i := (\hat{h}_i, \hat{u}_i, f_i)$. roof of $\log_h \bar{h}_i = \log_{g_T} \bar{u}_i$
$\bar{\mathsf{ct}}_{i,0} := h$.
$\bar{\mathsf{ct}}_{i,1} := (\bar{h}_i, \pi_i)$, $\bar{\mathsf{ct}}_{i,2} :$
Return $\bar{\mathsf{ct}}_i = (\bar{\mathsf{ct}}_{i,0}$ $_{i,2})$.

Proof Generation: Prove that $\bar{ct}_i$ is generated by the correct token θ_i

- **Noninteractive Zero Knowledge Proof**

$\mathsf{msg}'/\perp \leftarrow \mathsf{Retrieve}(sk_{\mathcal{R}} = \beta, \{(\ell_j, \bar{\mathsf{ct}}_{\ell_j})\}_{j\in[n']})$:

If $n' \le t$: Return $\perp$.
Parse $\bar{\mathsf{ct}}_{\ell_j} = (\bar{\mathsf{ct}}_{\ell_j,0}, \bar{\mathsf{ct}}_{\ell_j,1}, \bar{\mathsf{ct}}_{\ell_j,2})$ for $j \in [n']$.
Filter set $\{(\ell_j, \bar{\mathsf{ct}}_{\ell_j})\}_{j\in[n']}$ to obtain subset $\mathcal{S} = \{\{(i_j, \bar{\mathsf{ct}}_{i_j})\}_{j\in[n'']}\}$ satisfying $n'' \ge t+1$

$$\bigwedge_{j,j'\in[n'']} \Big((\bar{\mathsf{ct}}_{i_j,0} = \bar{\mathsf{ct}}_{i_{j'},0}) \wedge (\bar{\mathsf{ct}}_{i_j,2} = \bar{\mathsf{ct}}_{i_{j'},2}) \Big): \qquad (\star)$$

$\mathcal{V} := \emptyset$.
For $j \in [n'']$
Parse $\bar{\mathsf{ct}}_{i_j,0} = h, \ \bar{\mathsf{ct}}_{i_j,1} = (\bar{h}_{i_j}, \pi_{i_j} = (\hat{h}_{i_j}, \hat{u}_{i_j}, f_{i_j}))$.
$//\ 0/1 \leftarrow \mathsf{ZKVerify}((h, \bar{h}_{i_j}, g_T, \bar{u}_{i_j}), \pi_{i_j})$

$\bar{u}'_{i_j} := \prod_{\ell=0}^{t} (v_\ell)^{i_j^\ell}$.
$e_{i_j} := H(h, g_T, \bar{h}_{i_j}, \bar{u}'_{i_j}, \hat{h}_{i_j}, \hat{u}_{i_j})$.
If $(h^{f_{i_j}} = \hat{h}_{i_j} \cdot \bar{h}_{i_j}^{e_{i_j}}) \wedge (g_T^{f_{i_j}} = \hat{u}_{i_j} \cdot \bar{u}'^{e_{i_j}}_{i_j})$: $\qquad (\star\star)$
$\mathcal{V} := \mathcal{V} \cup \{(i_j, \bar{h}_{i_j})\}$.

// end of proof verification of $\log_h \bar{h}_{i_j} = \log_{g_T} \bar{u}'_{i_j}$
If $|\mathcal{V}| \le t$: Return $\perp$.
Parse $\mathcal{V} = \{(j_0, \bar{h}_{j_0}), (j_1, \bar{h}_{j_1}), \ldots, (j_t, \bar{h}_{j_t}), \ldots\}$.
$k' := \Big(\prod_{i=0}^{t} \bar{h}_{j_i}^{\lambda_{j_i 0}}\Big)^{\beta^{-1}}$. $//\lambda_{j_i 0}$'s are Lagrange coefficients
$\mathsf{msg}' \leftarrow \mathsf{SE.Dec}(k', \bar{\mathsf{ct}}_{j_1,2})$. $//\bar{\mathsf{ct}}_{j_1,2} = \bar{\mathsf{ct}}_{j_2,2} = \ldots = \bar{\mathsf{ct}}_{j_t,2}$
Return msg'.

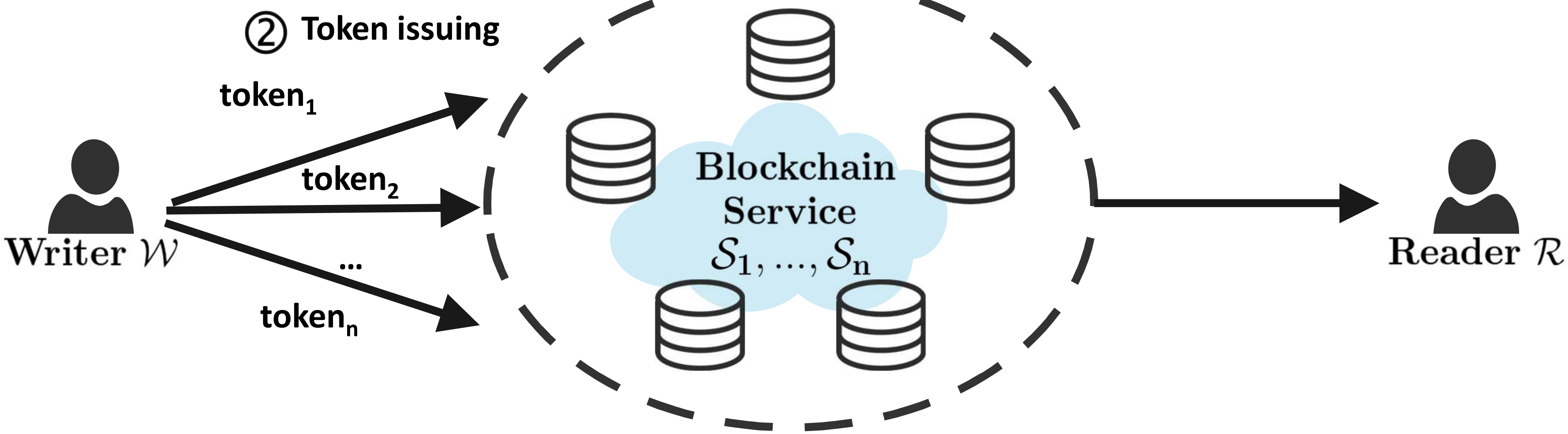
Proof Verification: Verify that $\bar{ct}_i$ is generated by the correct token θ_i

A quick recap

- **(t, n) - Verifiable Secret Sharing**

Key generation
$(pk_W, sk_W) \leftarrow$ KeyGen1

Key generation
$(pk_R, sk_R) \leftarrow$ KeyGen2



① **Token authorization**
$(token_1, \ldots, token_n) \leftarrow$ Authorize

③ **Token certification**
$0/1 \leftarrow$ Certify($token_i$) **Token verifiability**

A quick recap

- (t,n) – Threshold Proxy Re-Encryption
- Non-Interactive Zero Knowledge Proof

Writer $\mathcal{W}$

① Encrypt

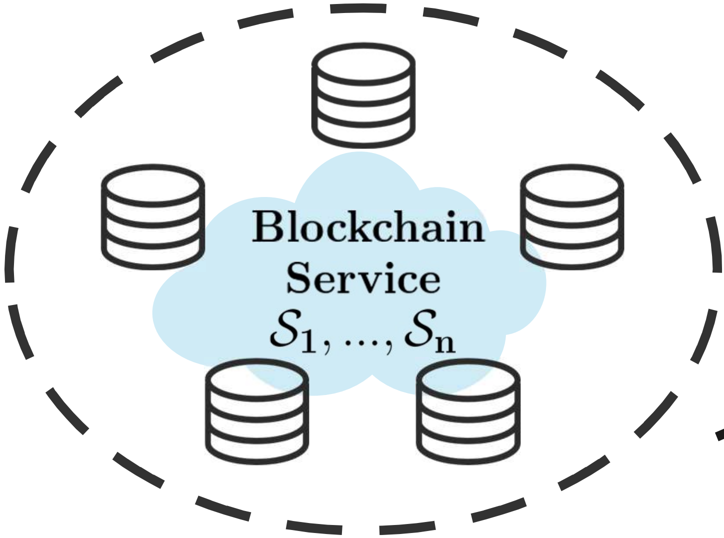
$ct \leftarrow$ Encrypt(pk_w, msg)

② Write

Ciphertext ct,
Access list acl

③ Consensus

Server S_1, ... S_n run consensus to agree on (ct, acl_w)



④ Convert

Each S_i checks if policy(acl_W,pk_R)=1.
If so, convert ct to $\bar{ct}_i$.

$\bar{ct}_i \leftarrow$ Convert($token_i$, ct, pk_R)

⑤ Read

$\bar{ct}_1$
$\bar{ct}_2$
...
$\bar{ct}_n$

Reader $\mathcal{R}$

⑥ Retrieve

Each S_i sends $\bar{ct}_i$ to R

msg'/$\perp \leftarrow$ Retrieve(sk_R, $\{\bar{ct}_i\}$)

t-cr-CPA-security

t-robustness

Contents

Conclusion

Confidentiality of blockchains enabled by ECA:

- ECA (t-robustness): Reader’s successful decryption in case of t corrupted replicas.
- ECA (Token verifiability): Replicas are able to verify tokens distributed by the writer, writers cannot cheat replicas.
- ECA (t-cr-CPA security): Confidentiality will be preserved in case of t colluding replicas.
- Causality preservation & Flexible token revocation

Thank you!