Protocol DKLs23.Sign

t-out-of-n threshold signing protocol from [DKLs23], realizing the standard ECDSA functionality with UC security for a group $\mathbb{G}(q, G)$. The protocol builds on DKG, Przs, $\mathsf{RVOLE}_{2,q}$, a Commitment scheme and a hash function H.

Players: Key share holders: $\{\mathcal{P}_i\}_{i\in[n]}$ holding $\{x_i\}_{i\in[n]}$ and public key Q
Quorum of signers: $\{\mathcal{P}_i\}_{i\in \mathrm{S}}$ for $\mathrm{S}\in[n]^t$ and $\mathrm{S}^*=\mathrm{S}\setminus\{i\}$
Inputs: A session identifier sid, and a message $\boldsymbol{m}$
Outputs: A partial signature σ_i per $\mathcal{P}_i$. A signature σ after aggregation

$\mathcal{P}_i$.Init()$\dashrightarrow(x_i, Q, \zeta_i)$
1: Run $(Q, x_i) \leftarrow$ DKG to obtain a public and a private key share
2: Run Przs.Setup1(), Przs.Setup2() and Przs.Setup3() to setup zero sharing
3: Run RVOLE.Setup() as Alice with $\mathcal{P}_k$ as Bob $\forall k\in[n]\setminus\{i\}$
4: Run RVOLE.Setup() as Bob with $\mathcal{P}_k$ as Alice $\forall k\in[n]\setminus\{i\}$

$\mathcal{P}_i$.Round1()$\dashrightarrow(R_i, \{c'_{ij}, \gamma_{ij}\}_{j\in \mathrm{S}^*})$
1: Sample $\phi_i \xleftarrow{\$} \mathbb{Z}_q$ as an inversion mask and $r_i \xleftarrow{\$} \mathbb{Z}_q$ as an instance key
2: $R_i \leftarrow r_i \cdot G$ as the public instance key
3: **for** $j \in \mathrm{S}^*$ **do**
4: Run $(c'_{ij}, w_{ij}) \leftarrow \mathsf{Commit}(i \mid\mid j \mid\mid sid \mid\mid R_i)$
5: Run $(\gamma_{ij}, b_{ij}) \leftarrow$ RVOLE.Round1() as Bob
6: $\mathsf{Send}(c'_{ij}, \gamma_{ij}) \to \mathcal{P}_j$
7: $\mathcal{F}^{Broadcast}(R_i)$

$\mathcal{P}_i$.Round2$(\{R_j, c'_{ji}, \gamma_{ji}\}_{j\in \mathrm{S}^*})\dashrightarrow(\{\boldsymbol{\mu}^{rnd2}_{ij}, \Gamma^u_{ij}, \Gamma^v_{ij}, b_{ij}, w_{ij}\}_{j\in \mathrm{S}^*}, R_i, P_i)$
1: Run $\zeta_i \leftarrow$ Przs.Sample() to get a zero share
2: $a_i \leftarrow \mathsf{ShamirToAdditive}(i, \mathrm{S}, x_i)$
3: $sk_i \leftarrow a_i + \zeta_i$ and $P_i \leftarrow sk_i \cdot G$ as refreshed instance key shares
4: **for** $j \in \mathrm{S}^*$ **do**
5: Run $(\boldsymbol{\mu}^{rnd2}, \boldsymbol{c}\equiv\{c^u, c^v\})_{ij} \leftarrow$ RVOLE.Round2$(\gamma_{ij}, \boldsymbol{a}=\{r_i, sk_i\})$ as Alice
6: $\Gamma^u_{ij} \leftarrow c^u_{ij} \cdot G$ and $\Gamma^v_{ij} \leftarrow c^v_{ij} \cdot G$
7: $\psi_{ij} \leftarrow \phi_i - b_{i,j}$
8: $\mathsf{Send}(\boldsymbol{\mu}^{rnd2}_{ij}, \Gamma^u_{ij}, \Gamma^v_{ij}, b_{ij}, w_{ij}, R_i) \to \mathcal{P}_j$
9: $\mathcal{F}^{Broadcast}(P_i)$

$\mathcal{P}_i$.Round3$(\boldsymbol{m}, \{\tilde{\boldsymbol{a}}_{ji}, \boldsymbol{\eta}_{ji}, \mu_{ji}, \Gamma^u_{ji}, \Gamma^v_{ji}, b_{ji}, w_{ji}, P_j\}_{j\in \mathrm{S}^*})\dashrightarrow\sigma_i$
1: **for** $j \in \mathrm{S}^*$ **do**
2: Run $\mathsf{Open}(j||i||sid||R_j, c'_{ji}, w_{ji})$, ABORT if it fails
3: Run $(\boldsymbol{d} \equiv \{d^u_{ij}, d^v_{ij}\}) \leftarrow$ RVOLE.Round3$(\boldsymbol{\mu}^{rnd2}_{ij}=\{\tilde{\boldsymbol{a}}, \boldsymbol{\eta}, \mu\})$ as Bob
4: Check if $b_{ji} \cdot R_j - \Gamma^u_{ji} \stackrel{?}{=} d^u_{ij} \cdot G$ otherwise ABORT
5: Check if $b_{ji} \cdot P_i - \Gamma^v_{ji} \stackrel{?}{=} d^v_{ij} \cdot G$ otherwise ABORT
6: Check if $\sum_{j\in \mathrm{S}} P_j \stackrel{?}{=} Q$, otherwise ABORT
7: $R \leftarrow \sum_{j\in \mathrm{S}} R_j$
8: $u_i \leftarrow r_i \cdot (\phi_i + \sum_{j\in \mathrm{S}^*} \psi_{ji}) + \sum_{j\in \mathrm{S}^*} (c^u_{ij} + d^u_{ij})$
9: $v_i \leftarrow sk_i \cdot (\phi_i + \sum_{j\in \mathrm{S}^*} \psi_{ji}) + \sum_{j\in \mathrm{S}^*} (c^v_{ij} + d^v_{ij})$
10: $w_i \leftarrow \mathsf{SHA2}(\boldsymbol{m}) \cdot \phi_i + (R_x) \cdot v_i$
11: **return** $\sigma_i = \{u_i, w_i\}$

Aggregate$(Q, \{u_j, w_j, R_j\}_{j\in \mathrm{S}})$--→$\sigma$

1: $R \leftarrow \sum R_j$ and $r \leftarrow R_x$
2: $s \leftarrow \frac{\sum w_j}{\sum u_j}$
3: $v \leftarrow (R_y \mod 2) + 2(R_x \overset{?}{\geq} q)$ as the recovery identifier $\in \mathbb{Z}_4$
4: **if** $(-s \mod q) \overset{?}{<} s$ **then** *(Normalize to "low s form")*
5: $\quad s \leftarrow (-s) \mod q$
6: $\quad v \leftarrow (v + 2) \mod 4$
7: Run $\mathsf{ECDSA.Verify}(Q, \boldsymbol{m}, \sigma = (r, s, v))$ to check if the signature is valid
return $\sigma = (r, s, v)$ as the signature

References

[DKLs23] Jack Doerner, Yashvanth Kondi, Eysa Lee, and A. shelat. Threshold ecdsa in three rounds. *Cryptology ePrint Archive*, 2023.