Decentralized Multi-Client Functional Encryption

Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, David Pointcheval

Université de Limoges,CNRS, Ecole Normale Supérieure, INRIA

December 5, 2018

Introduction

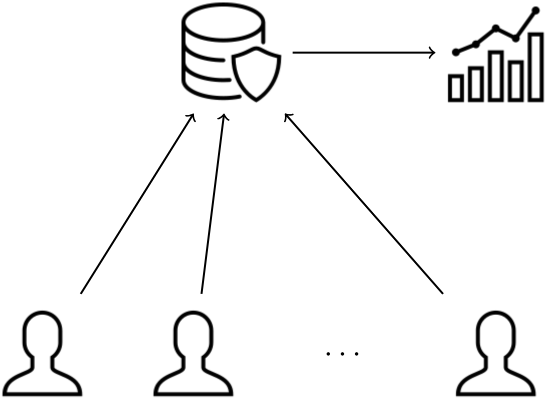
Definitions

Centralized scheme

Decentralized scheme

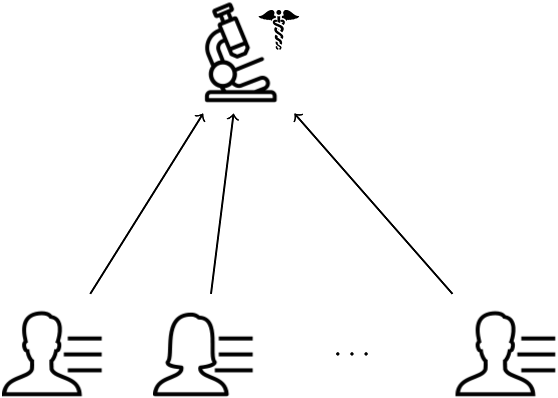
Context

Privacy and data analysis



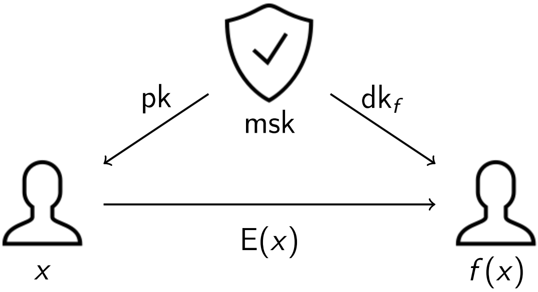
Context

Example: medical studies



Context

Functional Encryption (FE) [BSW11]:



IP-FE [ABDP15]: $f(\vec{x}) = \langle \vec{x}, \vec{y} \rangle$ for a given $\vec{y}$

Introduction

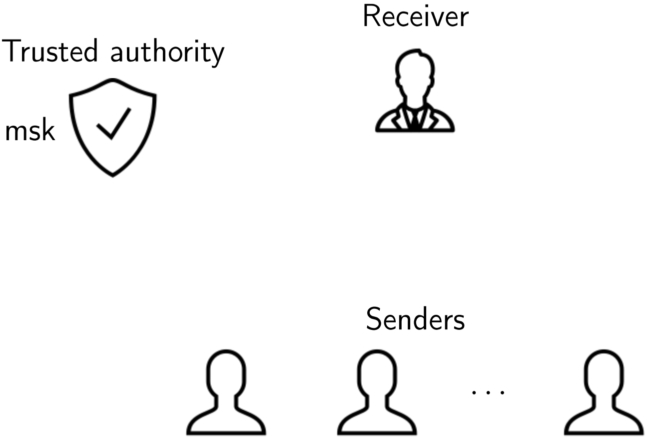
Definitions

Centralized scheme

Decentralized scheme

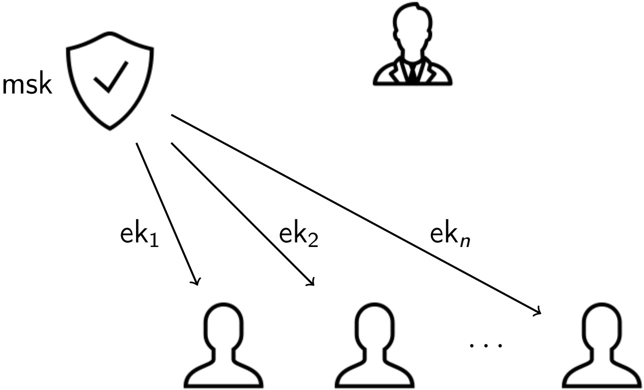
Description

Involved parties:



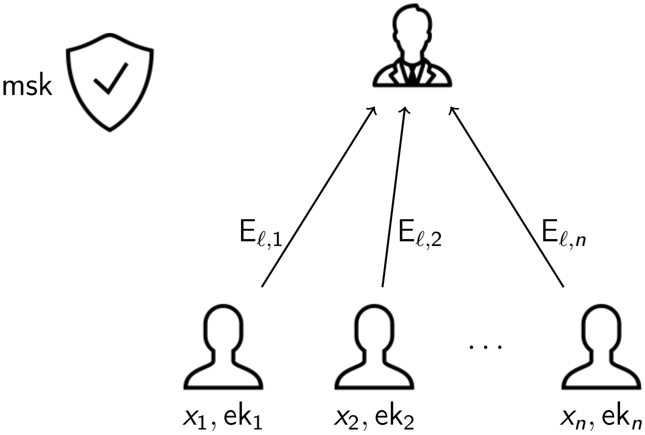
Description

SetUp: Sets msk, returns personal encryption keys ek_i



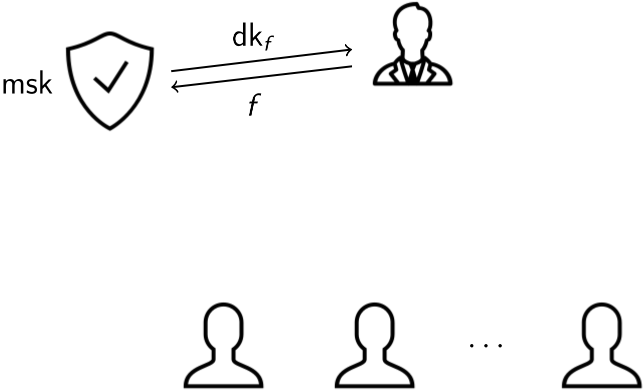
Description

Encrypt: Returns encrypted value $\mathsf{E}_{\ell,i}$



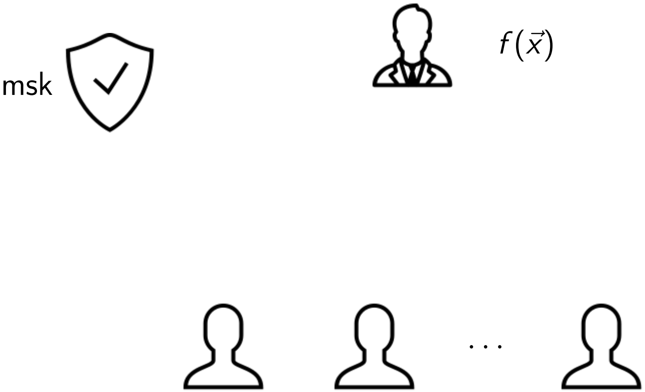
Description

DKeyGen: Given a function f, returns a decryption key dk_f



Description

Decrypt: Returns the value $f(\vec{x})$

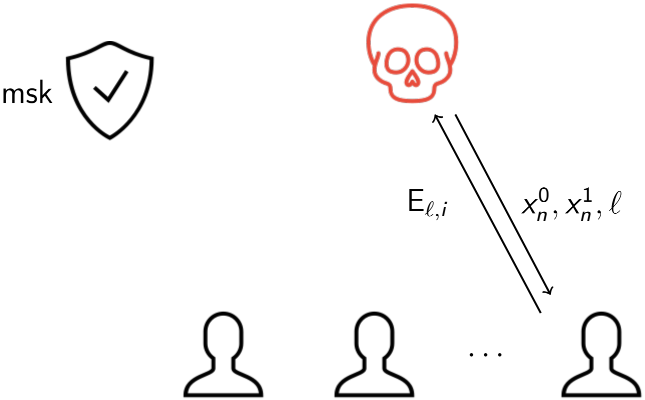


Security model

Malicious adversary

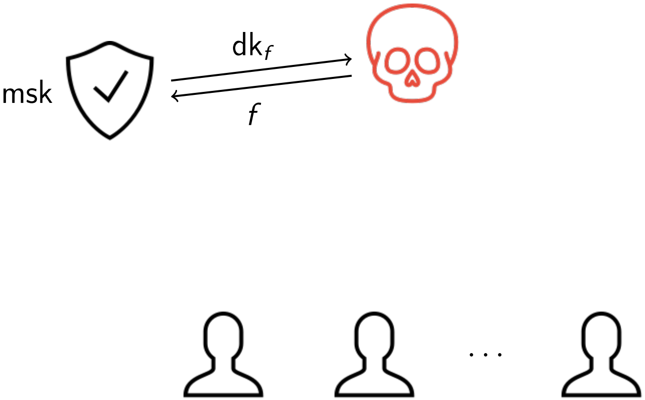
Security model

Encryption queries: QEncrypt$(i, (x_i^0, x_i^1), \ell)$



Security model

Decryption key queries: QDKeyGen(f)



Security model

Corruption of senders: QCorrupt(i)

Security model

Conditions of trivial win:

- $f(\vec{x}^0) \neq f(\vec{x}^1)$
- $x_i^0 \neq x_i^1$ for any corrupted sender i

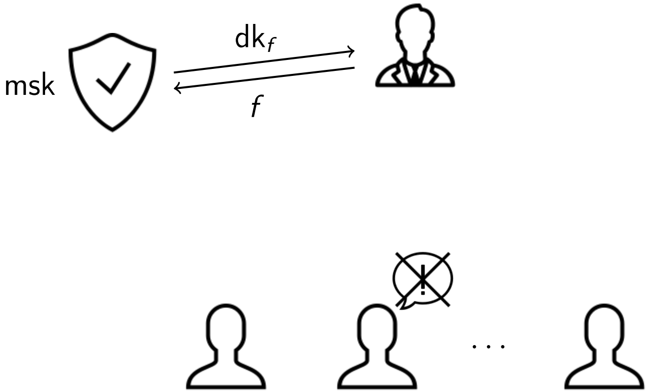
Security model

Variants:

- selective security
- static corruption

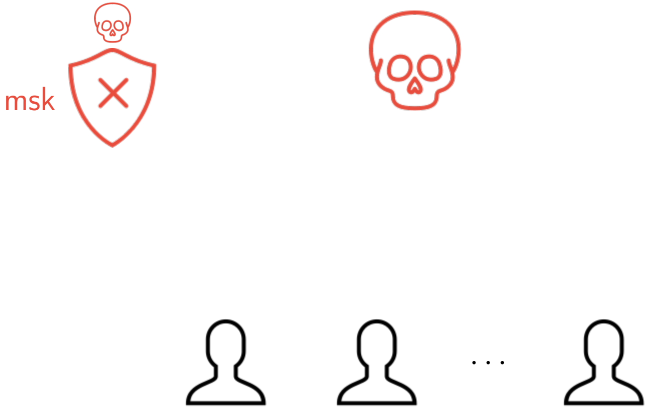
Why decentralization

No right management for senders



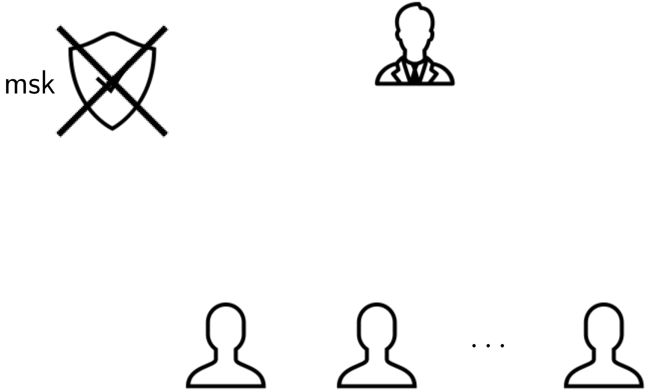
Why decentralization

Corruption of the authority



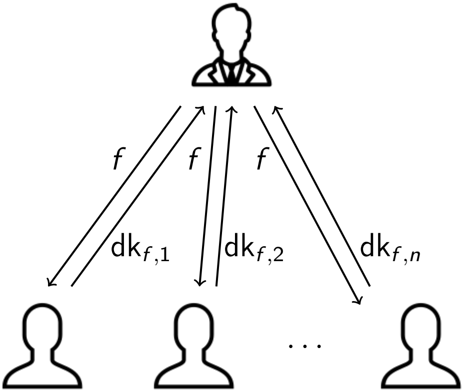
Why decentralization

Requires decentralization



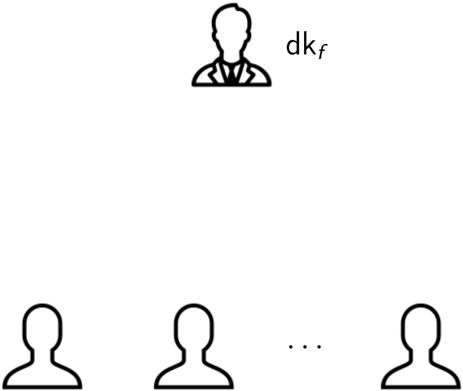
Impact on the definition

DKeyGenShare: returns a part $\mathsf{dk}_{f,i}$ of the future key dk_f



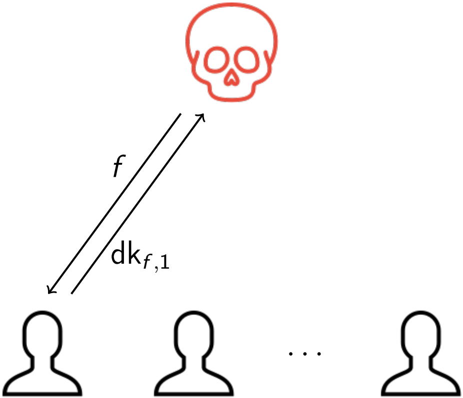
Impact on the definition

DKeyGenComb: Build the decryption key dk_f from $(\mathsf{dk}_{f,i})_i$



Impact on the security

Partial key queries: QDKeyGen(f, i)



Introduction

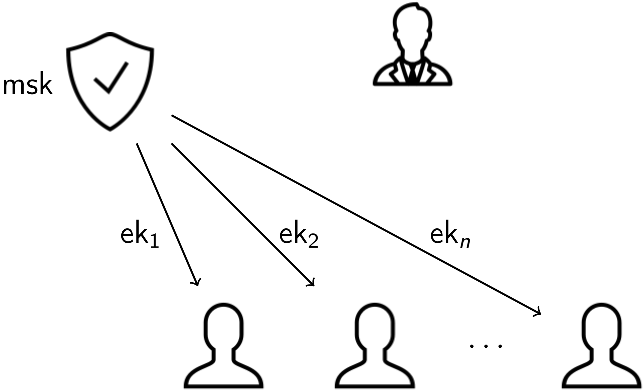
Definitions

Centralized scheme

Decentralized scheme

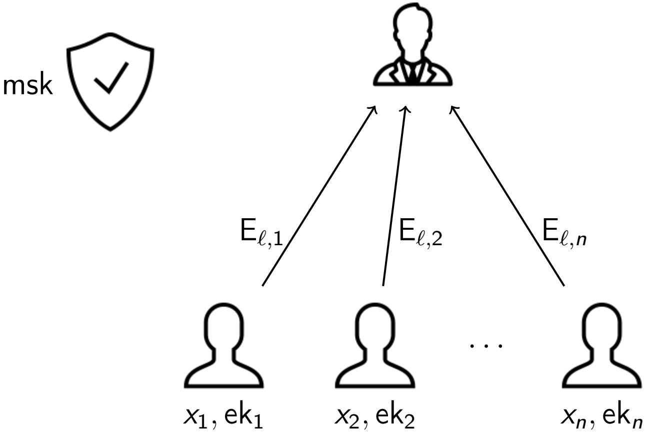
Description

SetUp(λ): $\mathsf{ek}_i \leftarrow \mathbb{Z}_p$, $\mathsf{msk} = (\mathsf{ek}_i)_i$, $\mathsf{pp} = (\mathbb{G}, \mathcal{H}$ onto $\mathbb{G})$



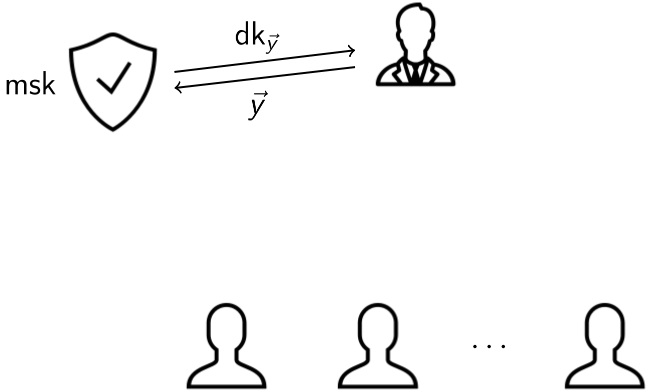
Description

Encrypt(ek_i,x_i,ℓ): $\mathsf{E}_{\ell,i} = \mathcal{H}(\ell)^{\mathsf{ek}_i} \cdot g^{x_i} \in \mathbb{G}$



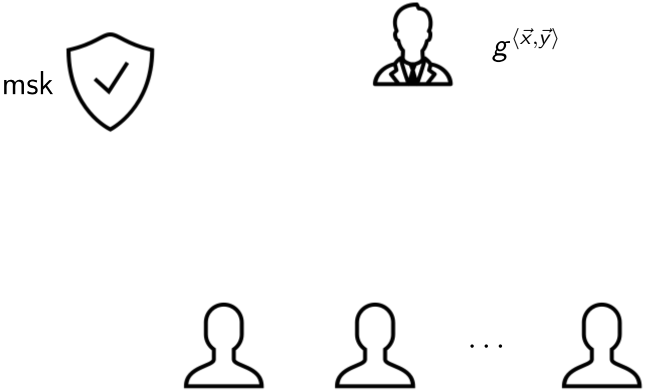
Description

DKeyGen(msk,$\vec{y}$): $\mathsf{dk}_{\vec{y}} = \langle \mathsf{msk}, \vec{y} \rangle = \langle (\mathsf{ek}_i)_i, \vec{y} \rangle \in \mathbb{Z}_p$



Description

Decrypt$(\mathsf{dk}_{\vec{y}}, \ell, \vec{\mathsf{E}}_\ell)$: $\prod_i \mathsf{E}_{\ell,i}^{y_i} \cdot \mathcal{H}(\ell)^{-\mathsf{dk}_{\vec{y}}}$



Description

Given $\mathsf{E}_{\ell,i} = \mathcal{H}(\ell)^{\mathsf{ek}_i} \cdot g^{x_i}$ and $\mathsf{dk}_{\vec{y}} = \langle \mathsf{msk}, \vec{y} \rangle$, the detail of the correctness is:

$$
\begin{aligned}
\alpha &= \prod_i \mathsf{E}_{\ell,i}^{y_i} \cdot \mathcal{H}(\ell)^{-\mathsf{dk}_{\vec{y}}} \\
&= \prod_i (\mathcal{H}(\ell)^{\mathsf{ek}_i} \cdot g^{x_i})^{y_i} \cdot \mathcal{H}(\ell)^{-\langle \mathsf{msk}, \vec{y} \rangle} \\
&= \mathcal{H}(\ell)^{\sum_i \mathsf{ek}_i y_i} \cdot g^{\sum_i x_i y_i} \cdot \mathcal{H}(\ell)^{-\langle \mathsf{msk}, \vec{y} \rangle} \\
&= g^{\langle \vec{x}, \vec{y} \rangle}
\end{aligned}
$$

Then solve the discrete logarithm of α

Security

- adaptive ciphertext queries
- adaptive corruption queries

Introduction

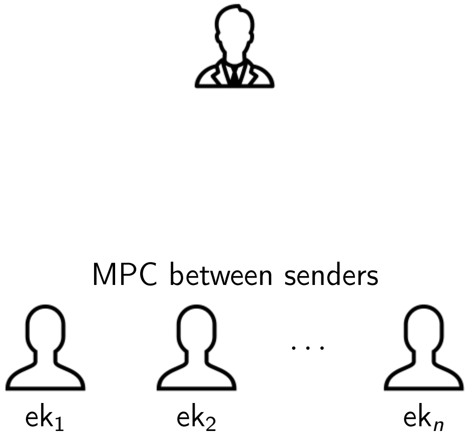
Definitions

Centralized scheme

Decentralized scheme

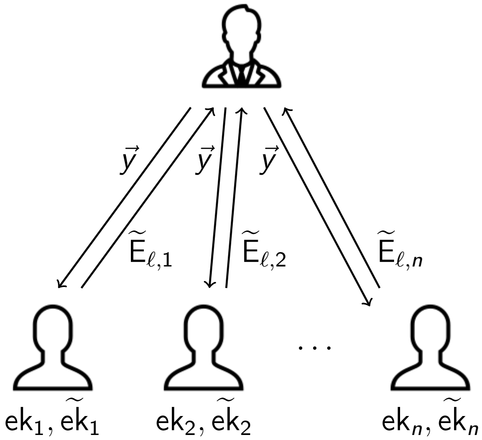
Idea

Problem: interactions for DKeyGen



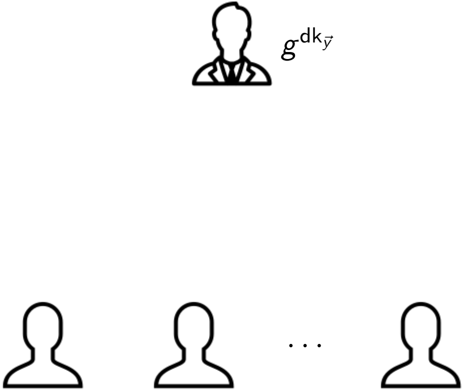
Idea

Solution: $\widetilde{\text{MCFE}}$ to build $\mathsf{dk}_{\vec{y}} = \langle (\mathsf{ek}_i y_i)_i, \vec{1} \rangle$



Idea

Specific problem: key as group element instead of scalar



Idea

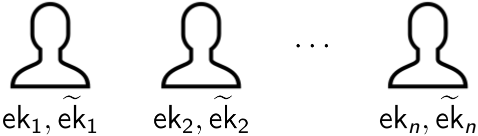
Specific solution: pairings

- message related MCFE in $\mathbb{G}_1$
- key related $\widetilde{\text{MCFE}}$ in $\mathbb{G}_2$

Description

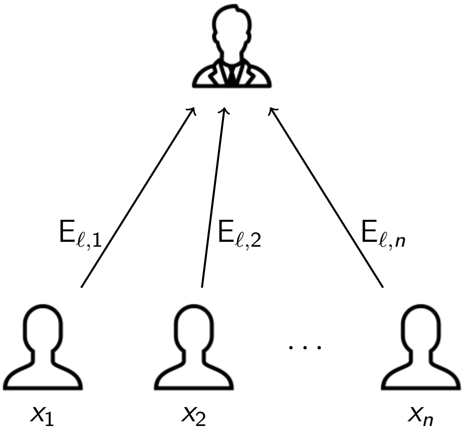
SetUp(λ): $\mathsf{ek}_i, \widetilde{\mathsf{ek}}_i \leftarrow \mathbb{Z}_p, \mathsf{sk}_i = (\mathsf{ek}_i, \widetilde{\mathsf{ek}}_i)$,
$\mathsf{pp} = (\mathbb{G}_1, \mathcal{H}_1, \mathbb{G}_2, \mathcal{H}_2, \mathbb{G}_T, e, \widetilde{\mathsf{dk}} = \sum_i \widetilde{\mathsf{ek}}_i)$

MPC between senders



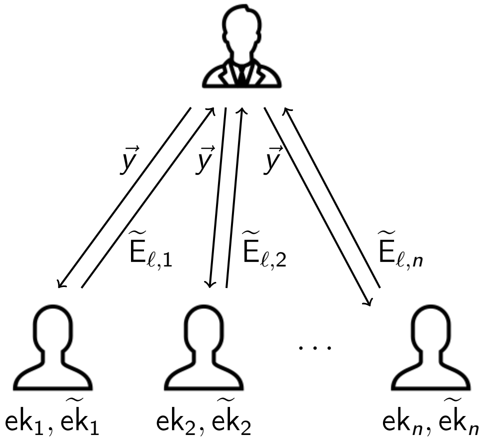
Description

Encrypt(ek_i,x_i,ℓ): $\mathsf{E}_{\ell,i} = \mathcal{H}_1(\ell)^{\mathsf{ek}_i} \cdot g_1^{x_i} \in \mathbb{G}_1$



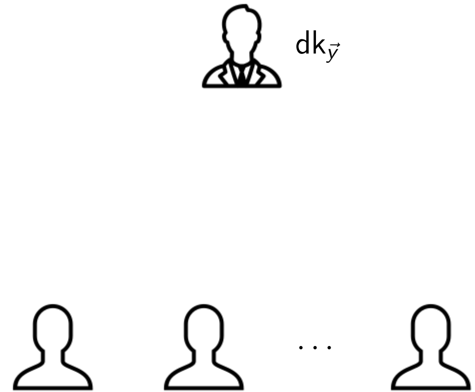
Description

DKeyGenShare(sk_i,$\vec{y}$): $\widetilde{\mathsf{E}}_{\ell,i} = \mathcal{H}_2(\vec{y})^{\widetilde{\mathsf{ek}}_i} \cdot g_2^{\mathsf{ek}_i y_i} \in \mathbb{G}_2$



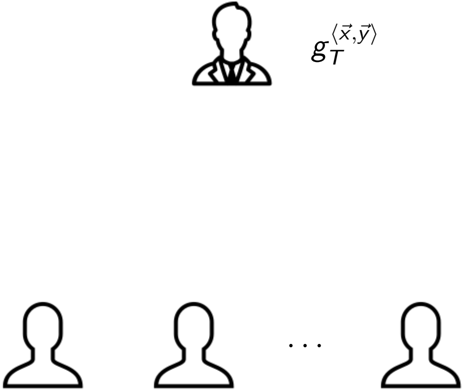
Description

$$\mathsf{DKeyGenComb}((\widetilde{\mathsf{E}}_{\ell,i})_i, \vec{y}):\ \mathsf{dk}_{\vec{y}} = \sum_i \mathsf{dk}_{\vec{y},i} \cdot \mathcal{H}_2(\vec{y})^{-\widetilde{\mathsf{dk}}} = g_2^{\sum_i \mathsf{ek}_i y_i} \in \mathbb{G}_2$$



Description

Decrypt$(\mathsf{dk}_{\vec{y}}, \ell, \vec{\mathsf{E}}_\ell)$: $e(\prod_i \mathsf{E}_{\ell,i}^{y_i}, g_2) \cdot e(\mathcal{H}_1(\ell), -\mathsf{dk}_{\vec{y}})$



Security

- adaptive ciphertext queries
- static corruption queries

Conclusion

In this paper we saw:

- MCFE primitive and scheme
- Security supporting corruptions
- Decentralization

Open questions:

- Security
- Number size
- Other function families

Conclusion

In this paper we saw:

- MCFE primitive and scheme
- Security supporting corruptions
- Decentralization

Open questions:

- Security
- Number size
- Other function families

Thank you !