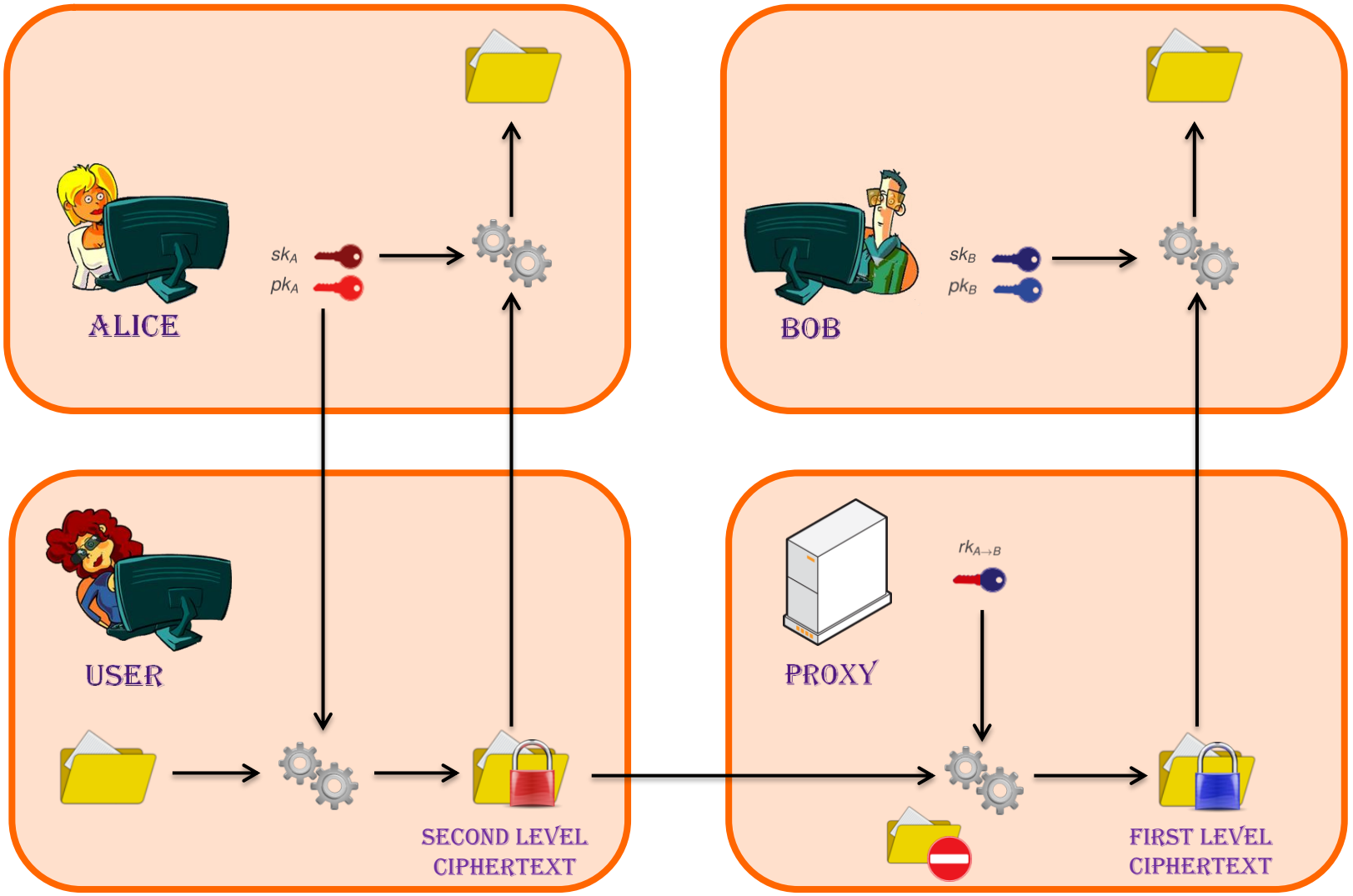
Combined Proxy Re-Encryption

Orange Labs, Applied Crypto Group,
Université de Caen Basse-Normandie, GREYC,

Sébastien Canard et **Julien Devigne**

Journées C2 2012, Dinard

Proxy Re-Encryption (PRE)



Simple Example (based on ElGamal)

- ElGamal encryption:
  - Introduced by ElGamal in 1984

Alice's secret key: x
Alice's public key: $X = g^x$

USER

$$m \rightarrow \begin{matrix} C_1 = X^r \\ C_2 = m.g^r \end{matrix}$$

ALICE

$$m = C_2.C_1^{1/x}$$

Simple Example (based on ElGamal)

- Re-Encryption key

$$\left.\begin{array}{l}\text{Alice's secret key: } x\\ \text{Bob's secret key: } y\end{array}\right\} \text{Re-encryption key: } R_{A\leftrightarrow B} = y/x$$

- Re-Encryption

$$C_1 = (g^x)^r$$
$$C_2 = m.g^r$$

$$\xrightarrow{R_{A\leftrightarrow B}}$$

$$C_1' = C_1^{R_{A\leftrightarrow B}} = (g^y)^r$$
$$C_2 = m.g^r$$

Usual Example (Libert-Vergnaud's PRE)

- Introduced in 2008

Alice's secret key: x
Alice's public key: $X = g^x$

u, v public parameters
OTS a one-time signature scheme

- Encryption / Decryption

USER

$$m \rightarrow$$

$$C_1 = svk$$
$$C_{2,A} = X^r$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

ALICE

$$e(C_{2,A}, u^{C_1}.v) == e(X, C_4)$$
$$V_{OTS}(C_1, \sigma, (C_3, C_4)) == 1$$
$$m = C_3 / e(C_{2,A}, g)^{1/x}$$

Usual Example (Libert-Vergnaud's PRE)

- Re-Encryption key

$$\left.\begin{array}{l}\text{Alice's secret key: } x \\ \text{Bob's public key: } Y = g^y\end{array}\right\} \text{Re-encryption key: } R_{A\to B} = (Y)^{1/x}$$

- Re-Encryption

ALICE

$$C_1 = svk$$
$$C_{A,2} = (g^x)^r$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

$R_{A\to B}$

BOB

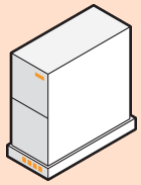
$$C_1 = svk$$
$$D_B = X^t = g^{xt}$$
$$E_B = R_{A\to B}^{1/t} = g^{y/xt}$$
$$F_B = C_{A,2}^t = g^{xrt}$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

Usual Example (Libert-Vergnaud's PRE)

Bob's secret key: y
Bob's public key: $Y = g^y$

u, v public parameters
OTS a one-time signature scheme

- Decryption of a re-encrypted ciphertext



PROXY

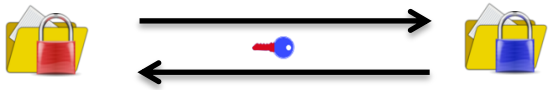
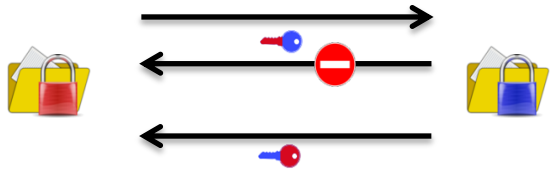
$$C_1 = svk$$
$$D_B = X^t = g^{xt}$$
$$E_B = R_{A\to B}^{1/t} = g^{y/xt}$$
$$F_B = C_{A,2}^t = g^{xrt}$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

BOB

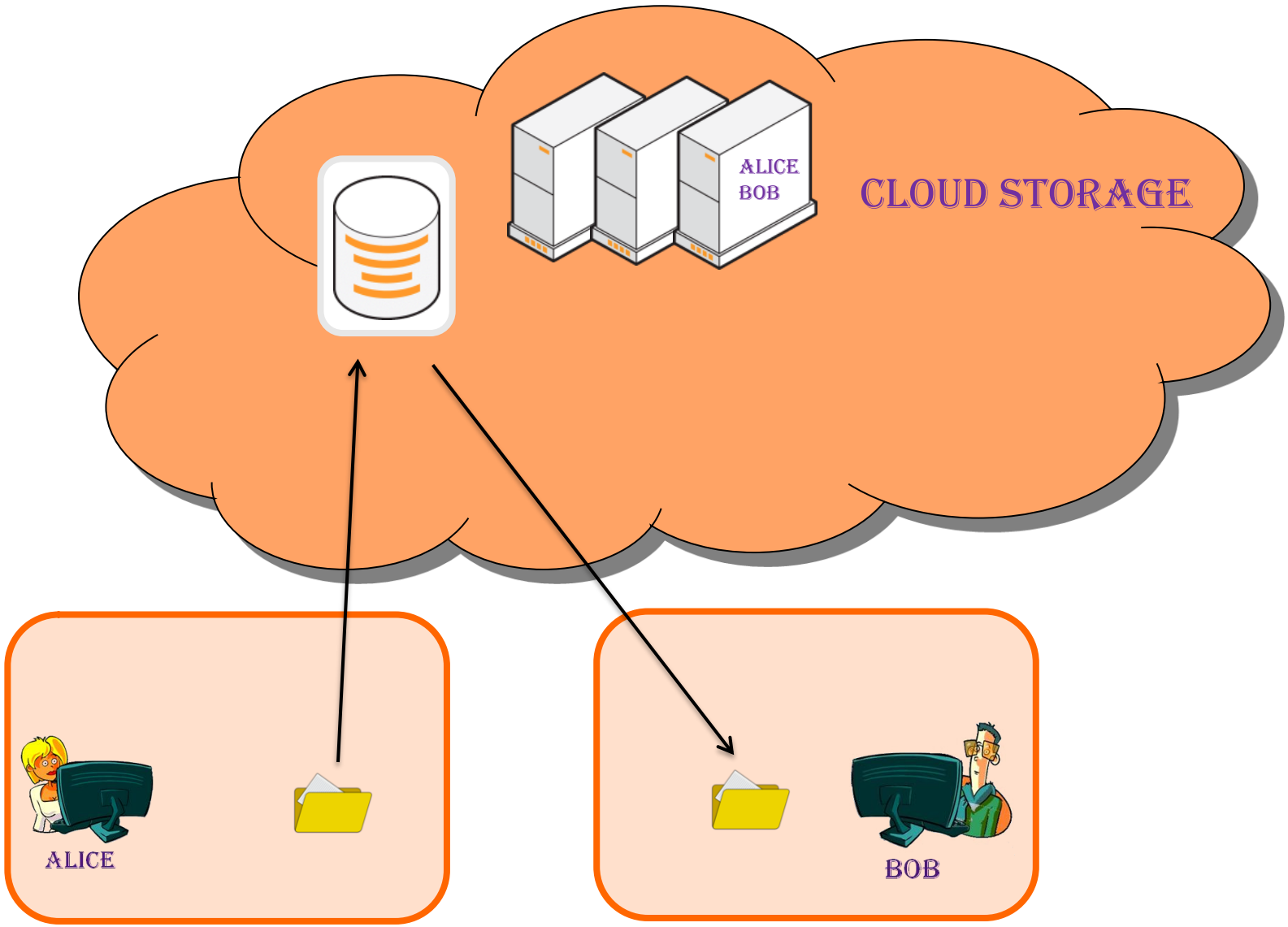
$$e(D_B, E_B) == e(Y, g)$$
$$e(F_B, u^{C_1}.v) == e(D_B, C_4)$$
$$V_{OTS}(C_1, \sigma, (C_3, C_4)) == 1$$

$$m = C_3 / e(E_B, F_B)^{1/y}$$

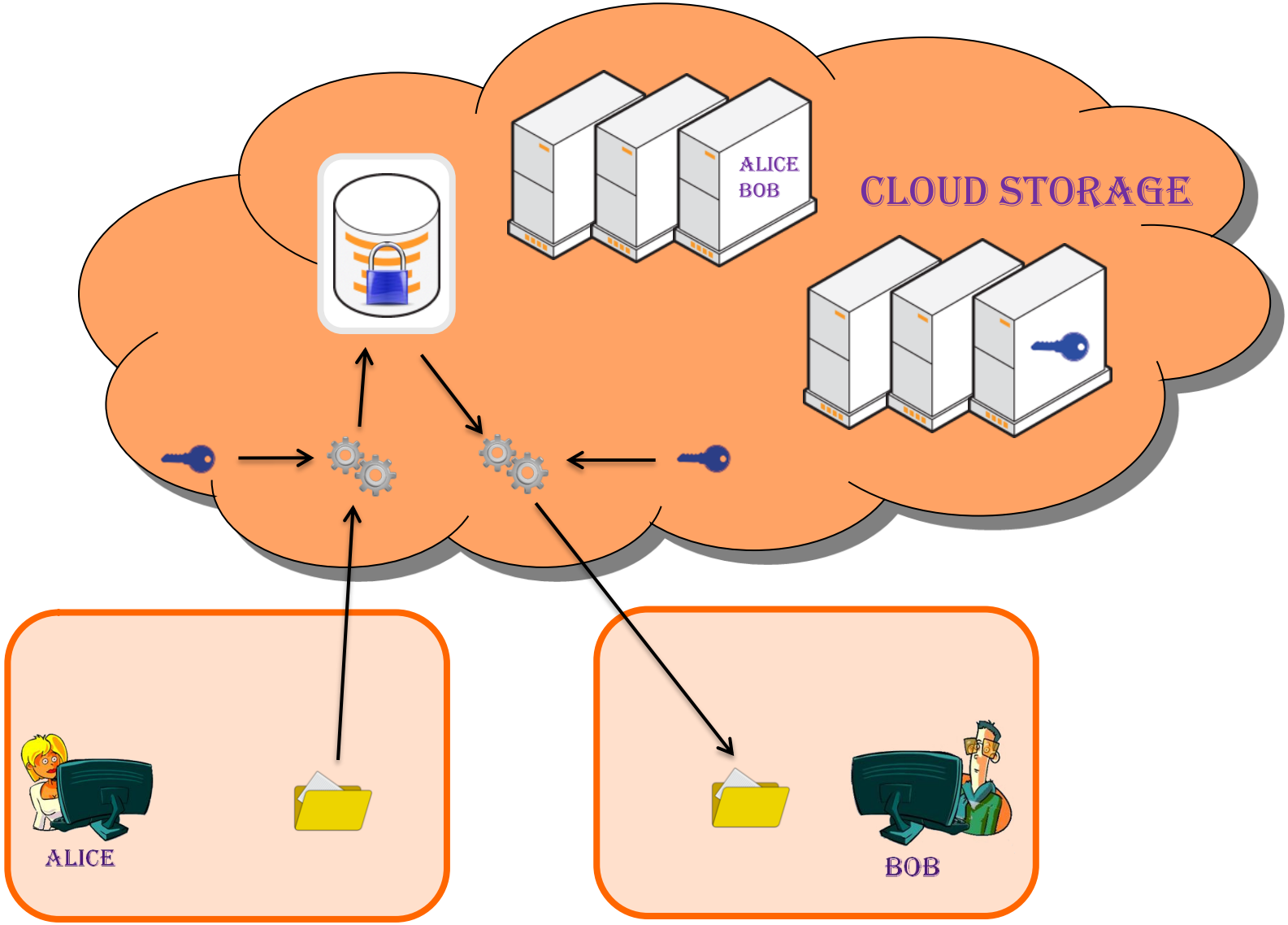
Characteristics

- Bidirectional  $R_{A \leftrightarrow B}: sk_A \longleftrightarrow sk_B$
- Unidirectional  $R_{A \rightarrow B}: sk_A \longrightarrow pk_B$
- Multi-hop 
- Single-hop 
- In practice:
  - Bidirectional multi-hop / Unidirectional single-hop

Cloud Storage

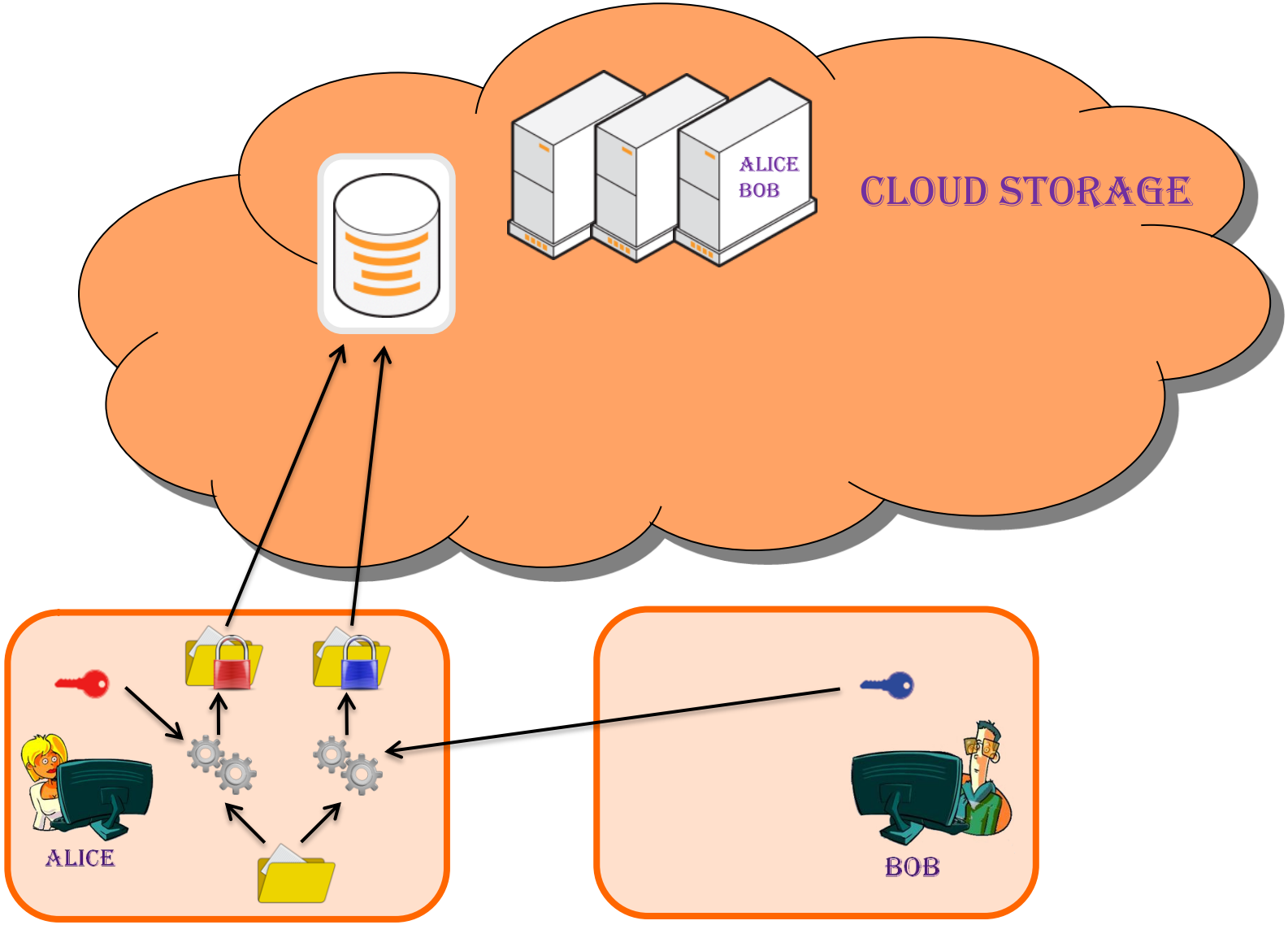


Secure Cloud Storage with Security of the Cloud

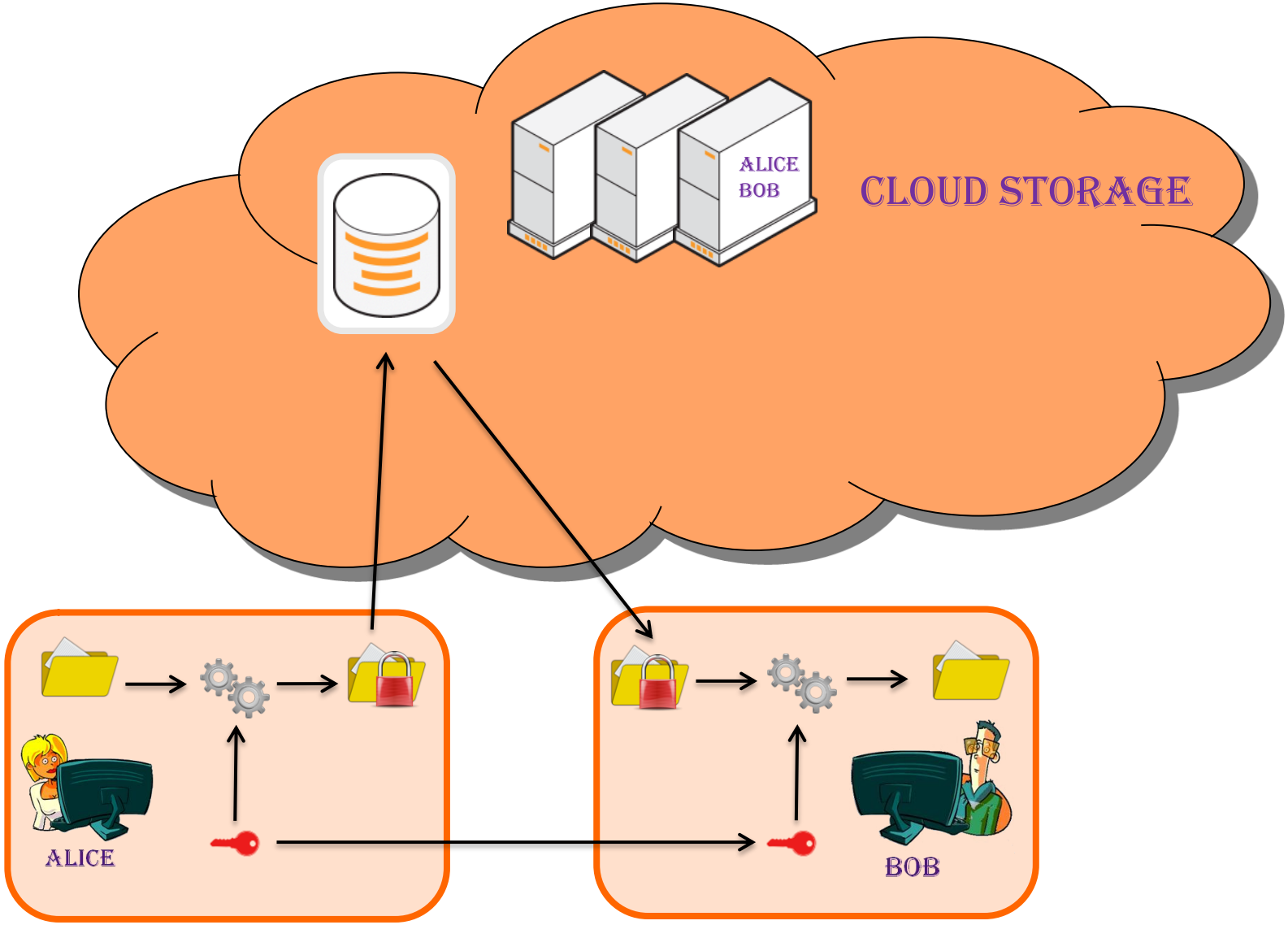


Combined Proxy Re-Encryption

Secure Cloud Storage with Duplication



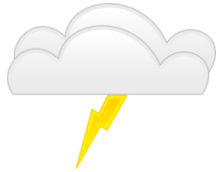
Secure Cloud Storage with Shared Key



Advantages / Drawbacks of the Cloud Storage

- Advantages
  - Saving device memory space
  - Accessibility from anywhere at anytime

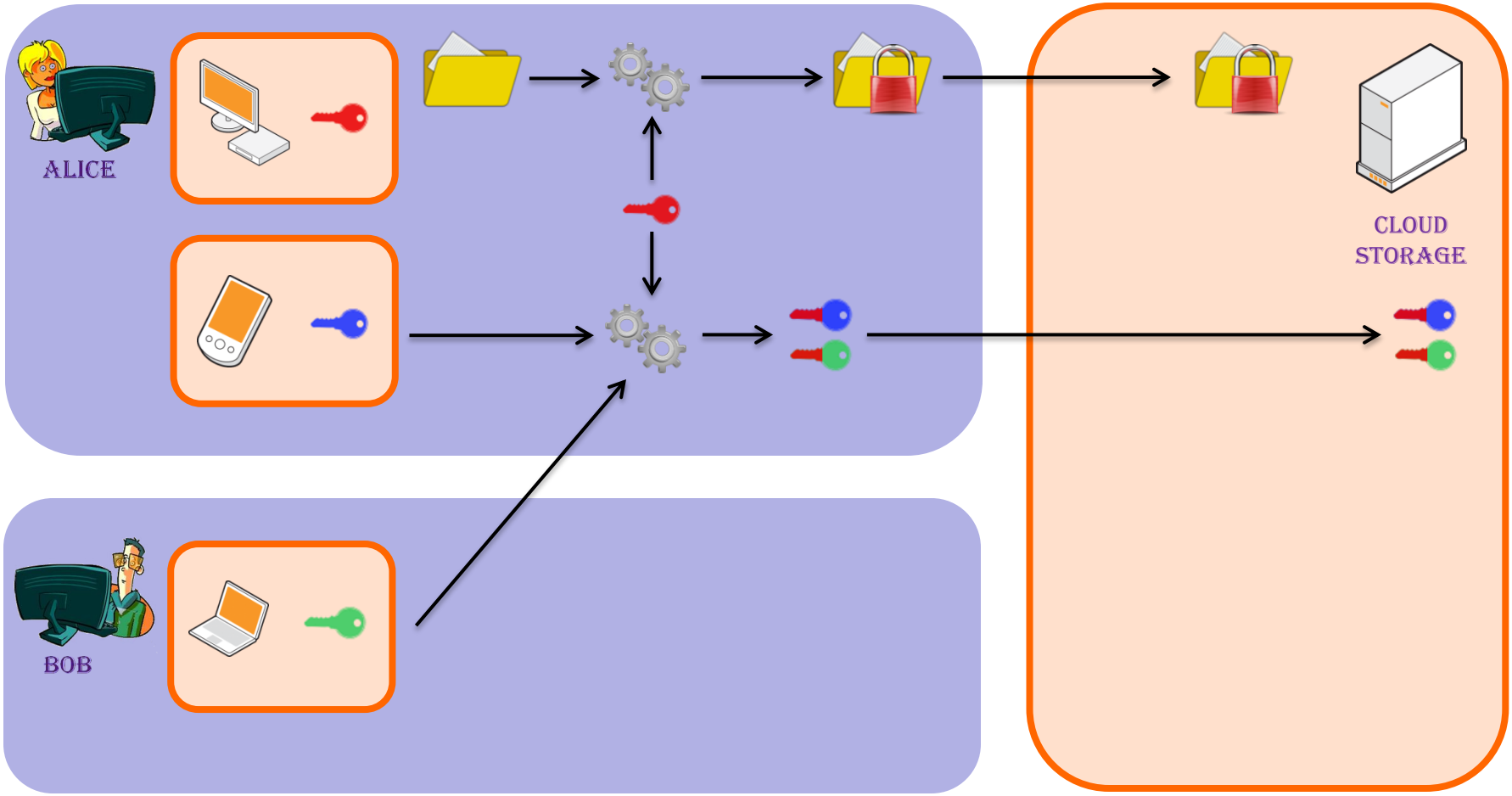
- Drawbacks depending on the solution
  - Security and no user's privacy, efficiency or confidentiality



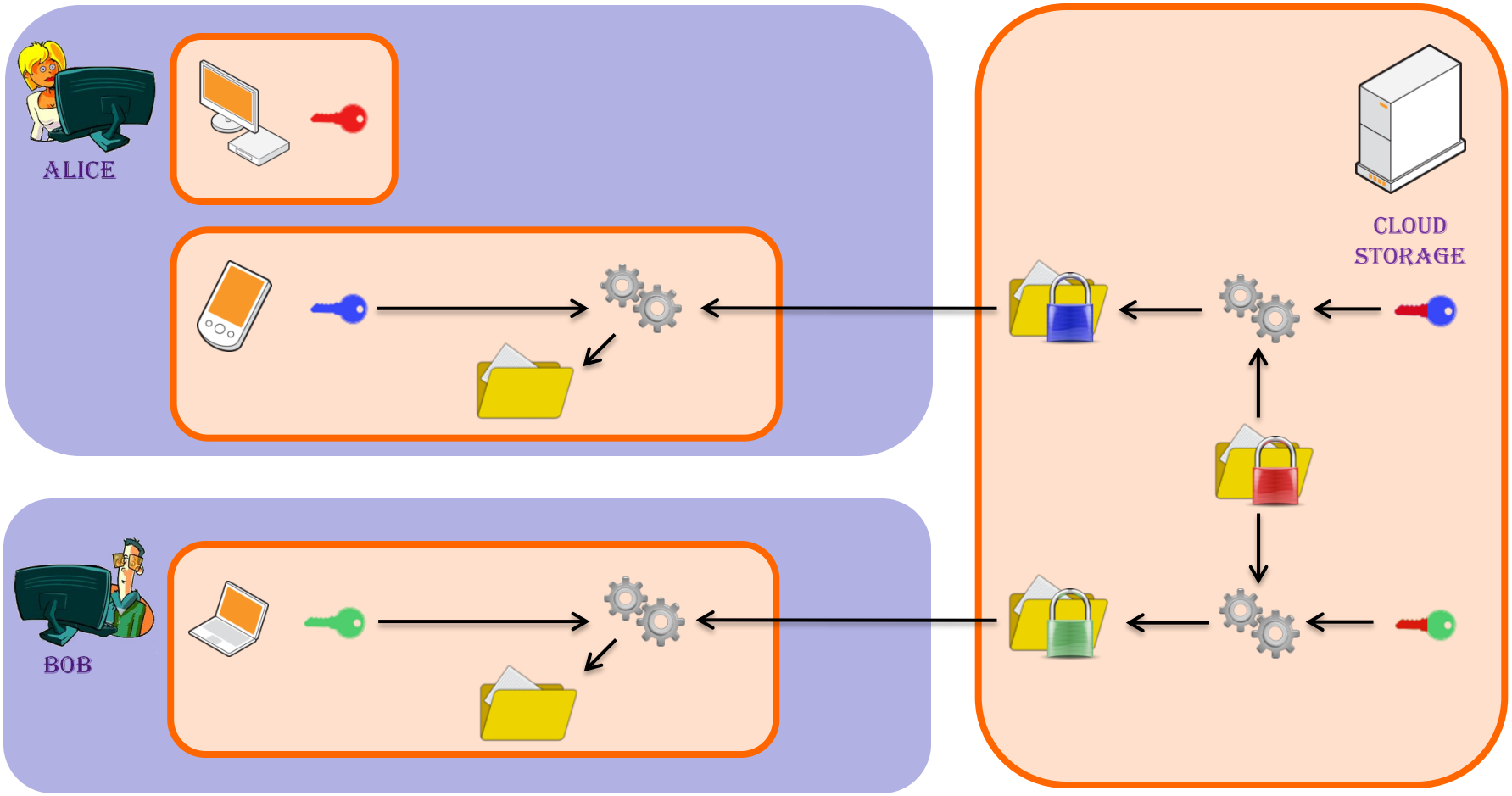
- Solution
  - Use a functional cryptographic primitive

➡ Proxy re-encryption primitive

Cloud Storage based on PRE / Store a data



Cloud Storage based on PRE / Recover a data



Advantages / Drawbacks of such a Cloud Storage

- Advantages
  - Privacy of the users
  - Security independent of the cloud

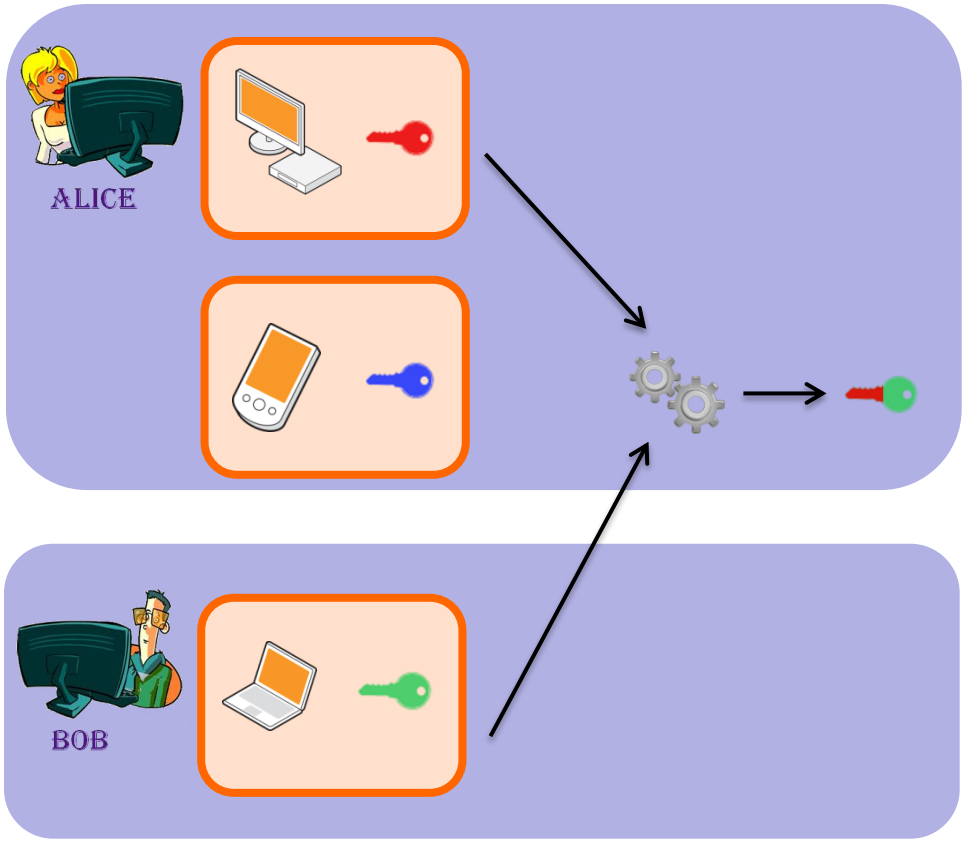
- Drawbacks
  - No control on the use of re-encryption keys

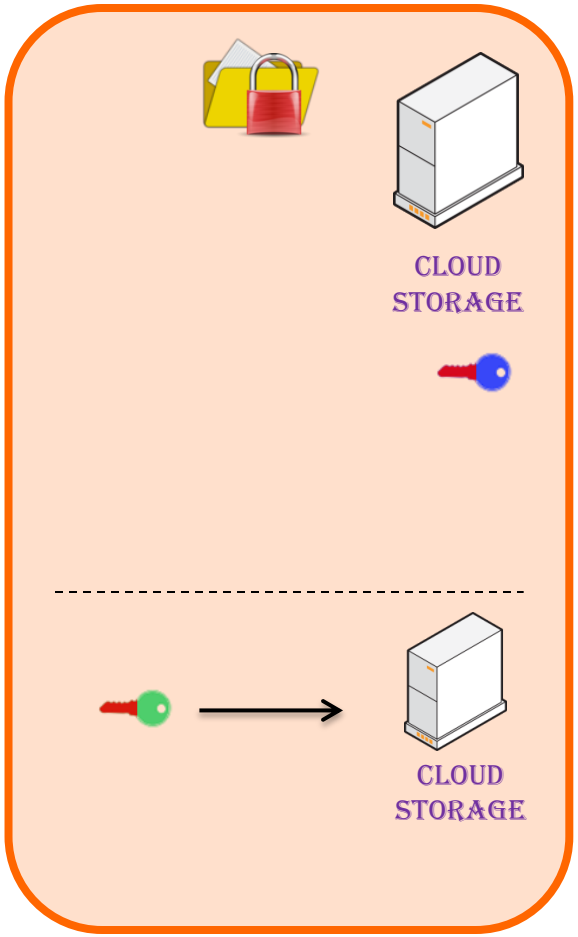
Bob has access to all Alice's data via a re-encryption done by the Cloud

- Solutions

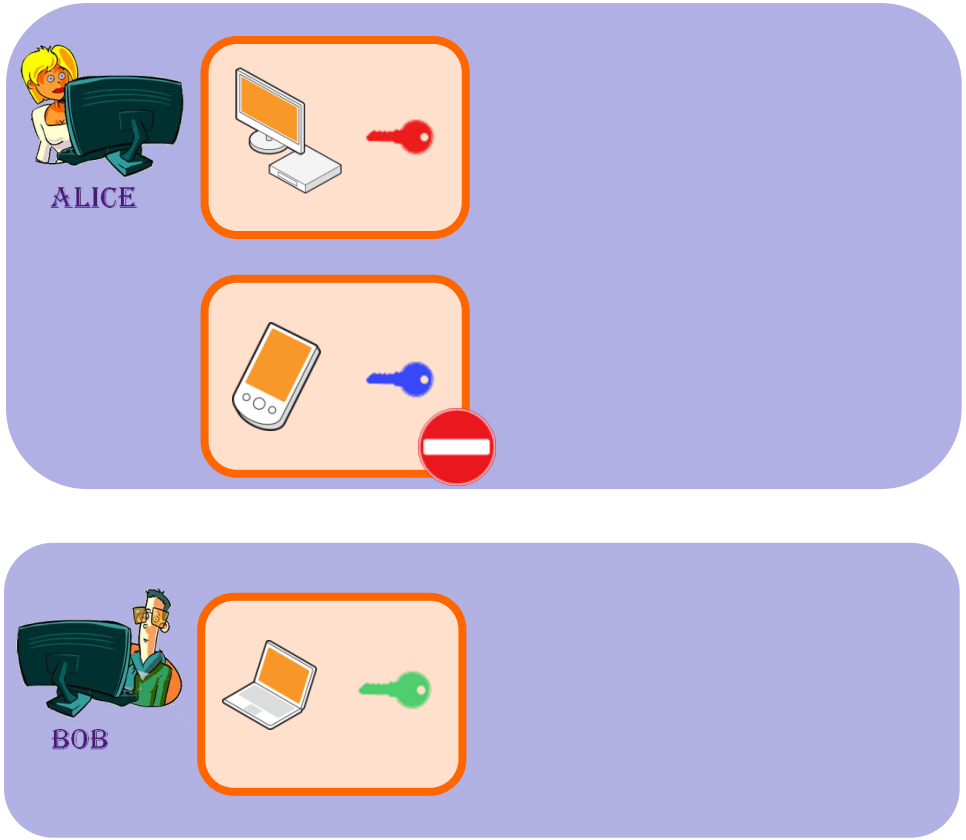
➡ Use PRE with more functionalities (Conditional-PRE)
(Not this talk)

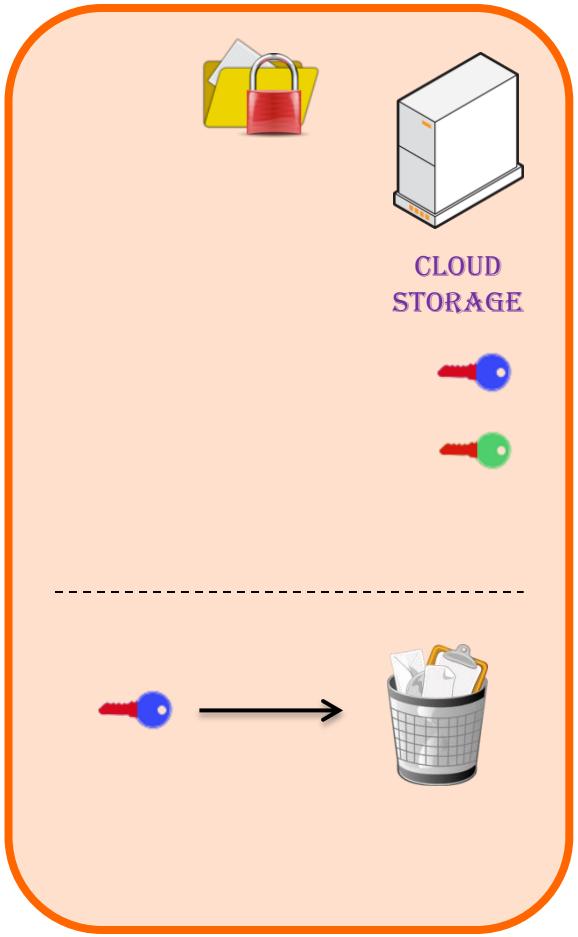
Add a new device (e.g. the green one)



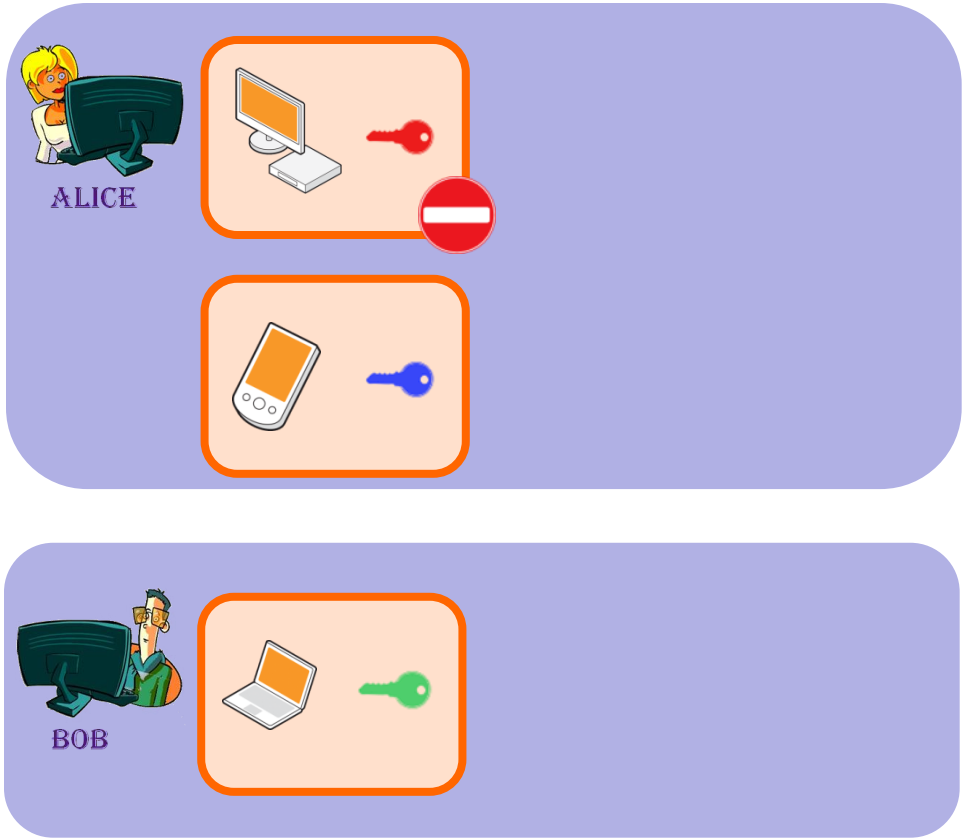


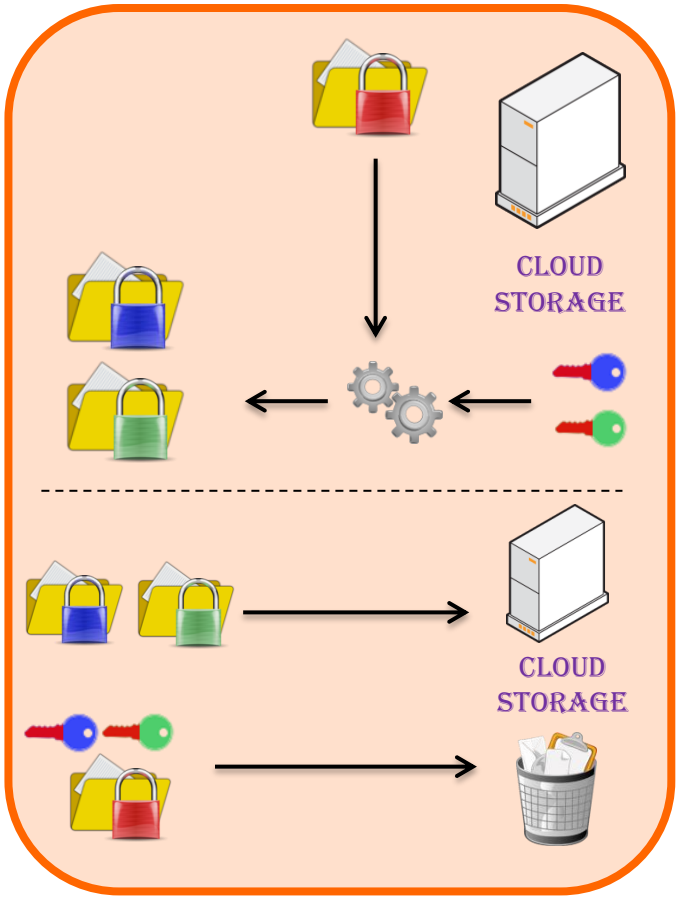
Delete the blue device (or the green one)





Delete the red device?





Advantages / Drawbacks of different PRE for this usecase

- Bidirectional multi-hop PRE
  - Multi-hop: possibility to add new devices even after deletion of the red device
  - Bidirectional: mutual trust between users

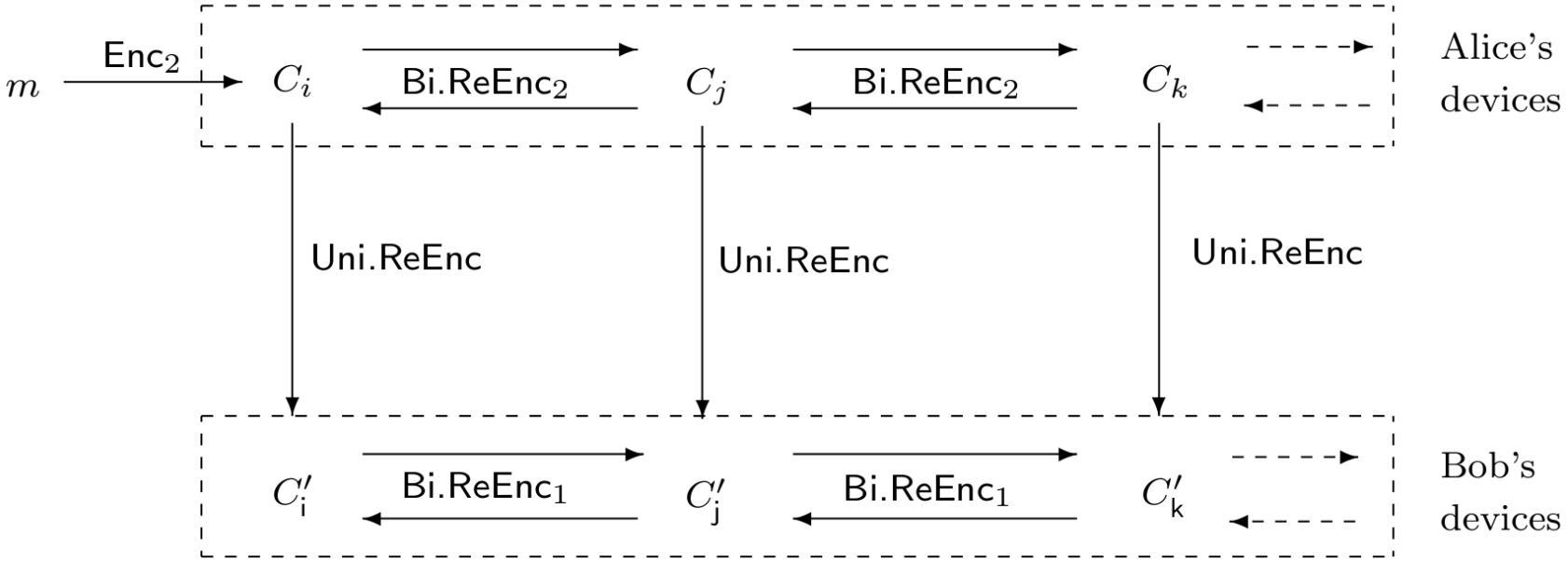
- Unidirectional single-hop PRE
  - Unidirectional: no mutual trust
  - Single-hop: no possibility to add new devices after deletion of the red device

- Ideally: unidirectional multi-hop PRE
  - No such secure scheme in practice

- Solution: combination of different PRE…

Idea of the Solution – Combined PRE (C-PRE)

- Use two kinds of re-encryption in the same scheme
  - Bidirectional multi-hop: for devices belonging to the same user
  - Unidirectional (and also single-hop): for devices belonging to different users



- Ideal for our problem!

Practical C-PRE based on Libert-Vergnaud's PRE

- Unidirectional Re-Encryption key

$$\left.\begin{array}{l}\text{Alice's secret key: } x \\ \text{Bob's public key: } Y = g^y\end{array}\right\} \text{ Re-encryption key: } R_{A\to B} = (Y)^{1/x}$$

- Unidirectional Re-Encryption

ALICE

$$C_1 = svk$$
$$C_{A,2} = (g^x)^r$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

$R_{A\to B}$ →

BOB

$$C_1 = svk$$
$$D_B = (g^x)^t$$
$$E_B = R_{A\to B}^{1/t} = g^{y/xt}$$
$$F_B = C_{A,2}^t = g^{xrt}$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

Practical C-PRE based on Libert-Vergnaud's PRE

- Bidirectional Re-Encryption key

$$\left.\begin{array}{l}\text{secret key of Alice's device 1: } x_1\\ \text{secret key of Alice's device 2: } x_2\end{array}\right\} \text{ Re-encryption key: } R_{A_1 \leftrightarrow A_2} = x_2/x_1$$

- Bidirectional Re-Encryption

ALICE_1

$$C_1 = svk$$
$$C_{A_1,2} = (g^{x_1})^r$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

$R_{A_1 \leftrightarrow A_2}$ →

ALICE_2

$$C_1 = svk$$
$$C_{A_2,2} = C_{A_1,2}^{R_{A_1 \leftrightarrow A_2}} = (g^{x_2})^r$$
$$C_3 = e(g,g)^r.m$$
$$C_4 = (u^{svk}.v)^r$$
$$\sigma = S_{OTS}(ssk, (C_3, C_4))$$

Practical C-PRE based on Libert-Vergnaud's PRE

- Bidirectional Re-Encryption key

$$\left.\begin{array}{l}\text{secret key of Bob's device 1: } y_1\\ \text{secret key of Bob's device 2: } y_2\end{array}\right\} \text{ Re-encryption key: } R_{B_1 \leftrightarrow B_2} = y_2/y_1$$

- Bidirectional Re-Encryption

BOB_1

$$\begin{aligned} C_1 &= svk \\ D_{B,1} &= (g^x)^t \\ E_{B,1} &= R_{A\to B}^{1/t} = g^{y_1/xt} \\ F_{B,1} &= C_{A,2}^t = g^{xrt} \\ C_3 &= e(g,g)^r.m \\ C_4 &= (u^{svk}.v)^r \\ \sigma &= S_{OTS}(ssk,(C_3,C_4)) \end{aligned}$$

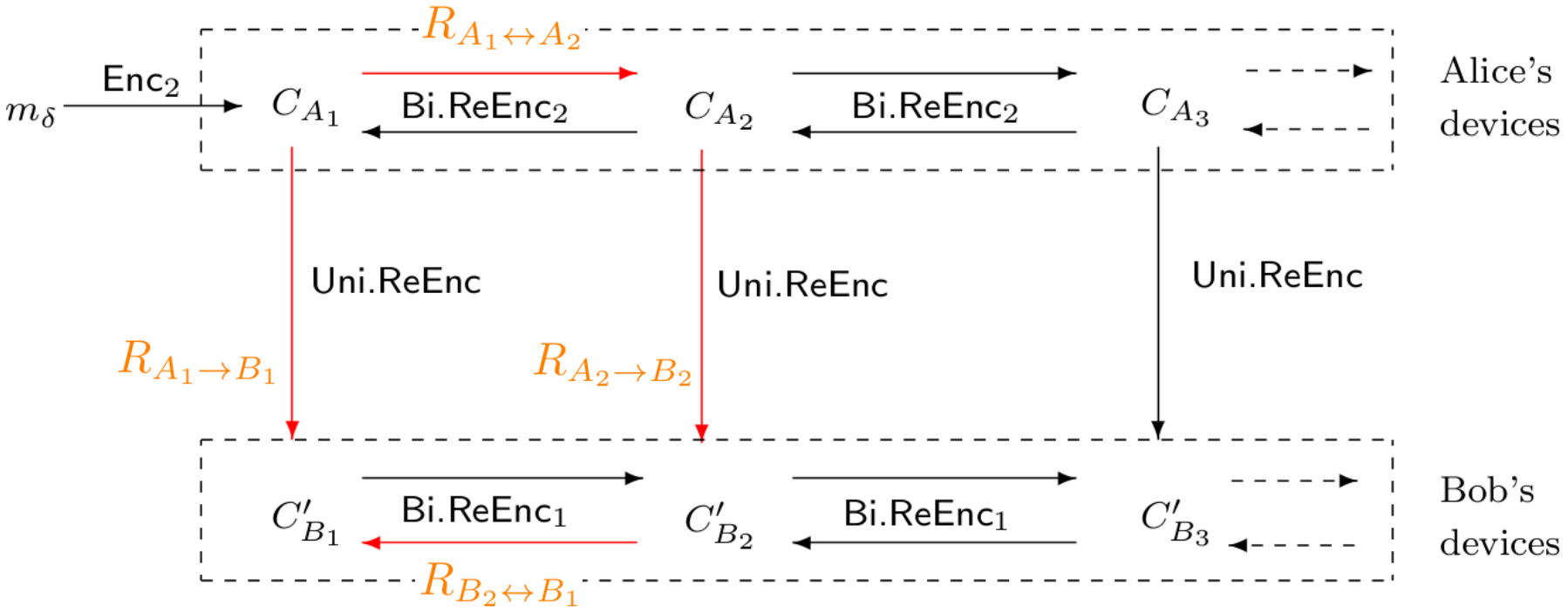
$R_{B_1 \leftrightarrow B_2}$ →

BOB_2

$$\begin{aligned} C_1 &= svk \\ D_{B,2} &= D_{B,1} = (g^x)^t \\ E_{B,2} &= E_{B,1}^{R_{B_1 \leftrightarrow B_2}} = g^{y_2/xt} \\ F_{B,2} &= F_{B,2} = g^{xrt} \\ C_3 &= e(g,g)^r.m \\ C_4 &= (u^{svk}.v)^r \\ \sigma &= S_{OTS}(ssk,(C_3,C_4)) \end{aligned}$$

Practical C-PRE based on Libert-Vergnaud's PRE

- Less re-encryption keys to compute per users



$$R_{A_1 \leftrightarrow A_2} = x_2/x_1$$
$$R_{A_2 \to B_2} = (g^{y_2})^{1/x_2} \Rightarrow R_{A_1 \to B_1} = R_{A_2 \to B_2}^{R_{A_1 \leftrightarrow A_2} \cdot R_{B_2 \leftrightarrow B_1}} = (g^{y_1})^{1/x_1}$$
$$R_{B_2 \leftrightarrow B_1} = y_1/y_2$$

Conclusion

- PRE useful to realize a cloud storage with confidentiality of data
- C-PRE add functionality to PRE
  - useful for the management of devices in a cloud storage
  - without modifying the efficiency of PRE
  - less re-encryption keys to compute
- Future work
  - Mix C-PRE and others PRE (e.g. Conditional-PRE)

Thanks

Comments/Questions?

université de Caen Basse-Normandie

orange™