Cryptography Engineering

- Lecture 8 (Dec 10, 2025)

- Case study: **E2EE-secure messaging - 2**
  - Forward/Backward Secrecy
  - Diffie-Hellman Ratchet

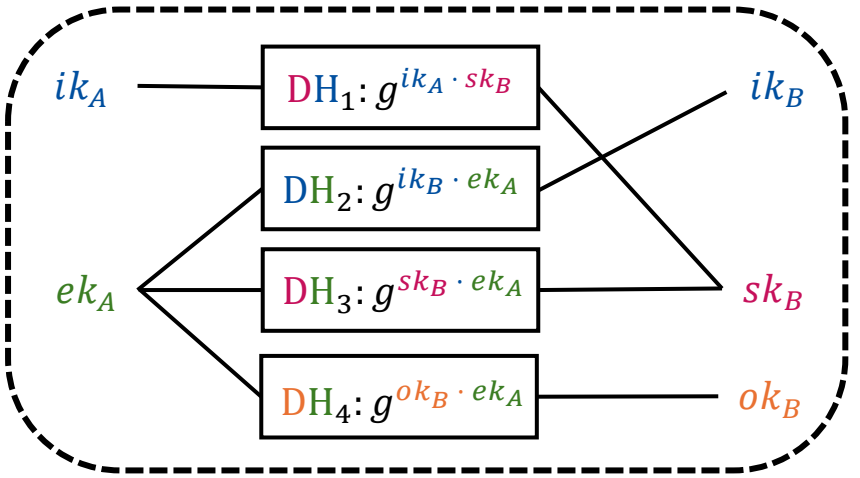
The X3DH Protocol

- How the X3DH protocol computes a shared secret…

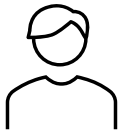
Alice

X3DH_Key_Alice(ik_A, ek_A, IPK_B, SPK_B, OPK_B)

1. $DH_1 = SPK_B^{ik_A}$
2. $DH_2 = IPK_B^{ek_A}$
3. $DH_3 = SPK_B^{ek_A}$
4. $DH_4 = (OPK_B)^{ek_A}$
5. $SK_A = KDF(DH_1, DH_2, DH_3, DH_4)$



Bob

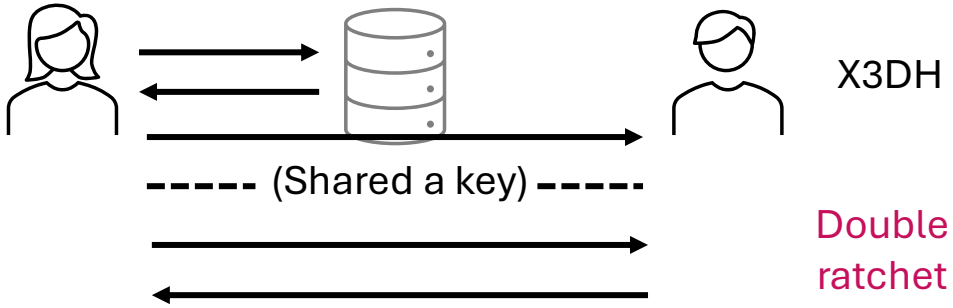


X3DH_Key_Bob(IPK_A, EPK_A, ik_B, sk_B, ok_B)

1. $DH_1 = IPK_A^{sk_B}$
2. $DH_2 = EPK_A^{ik_B}$
3. $DH_3 = EPK_A^{sk_B}$
4. $DH_4 = EPK_A^{ok_B}$
5. $SK_B = KDF(DH_1, DH_2, DH_3, DH_4)$

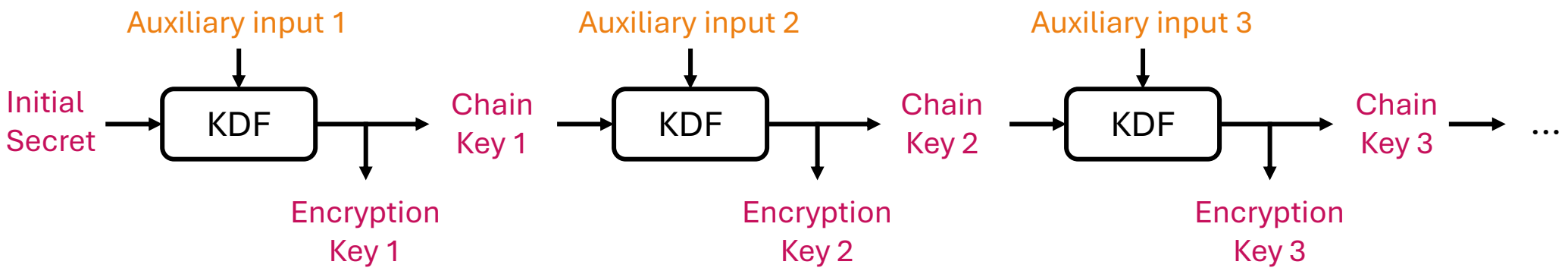
Double Ratchet

- After completing X3DH…
- … we use **Double Ratchet** to:
  - Encrypt messages + updates the shared key
  - ~~Encrypt messages using the same shared key~~
  - **Diffie-Hellman Ratchet** + **Symmetric-key Ratchet**
- Essential for forward/backward secrecy



Symmetric-key Ratchet

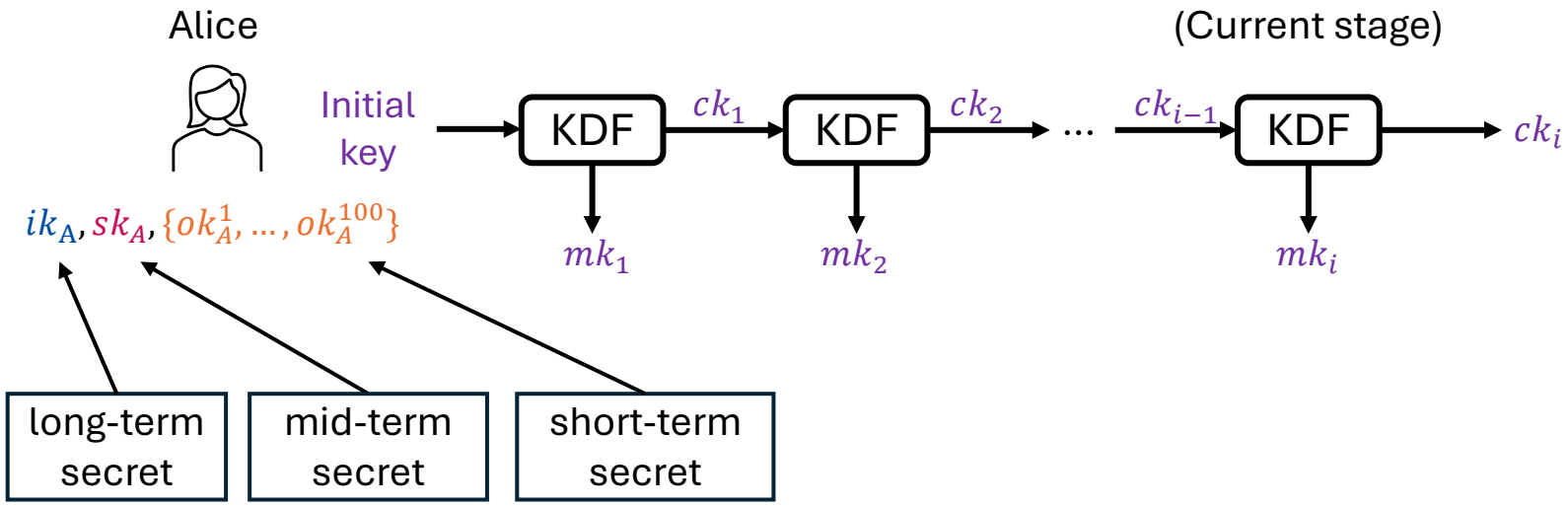
- KDF chain
  - KDF: Key derivation function



- Use Key Chain to encrypt messages

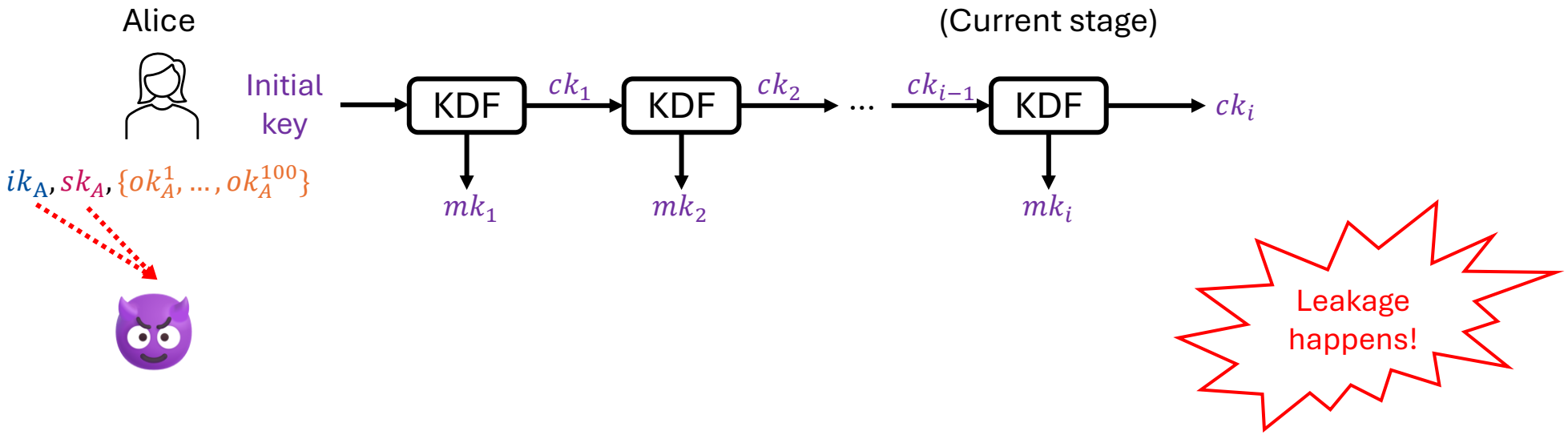
Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…



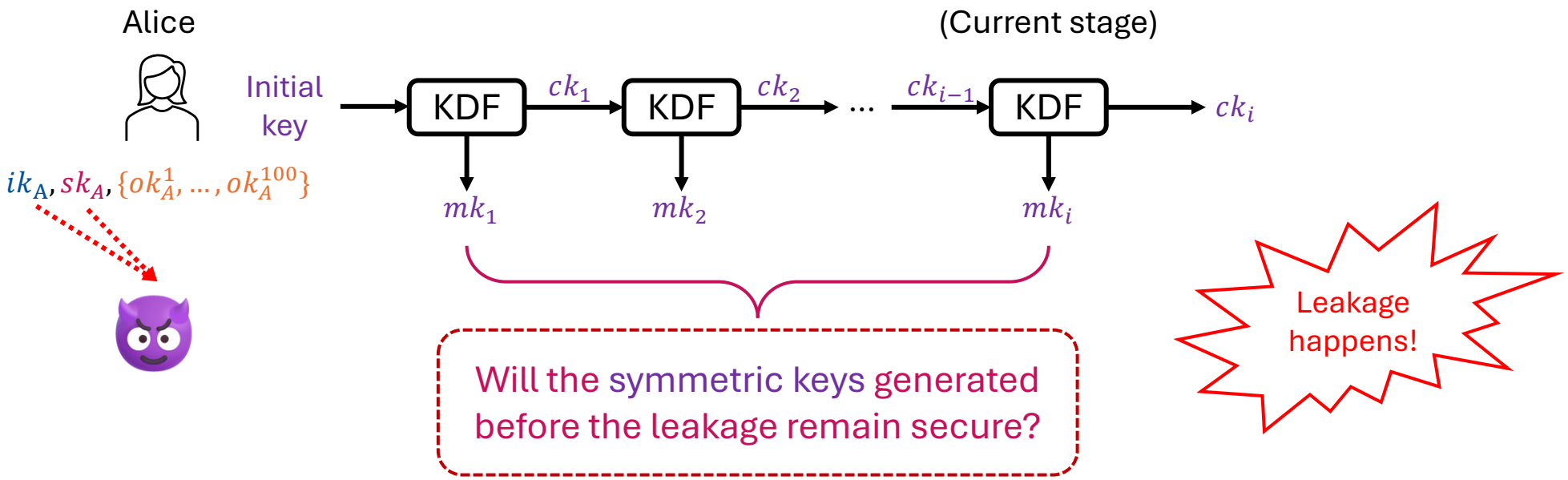
Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…



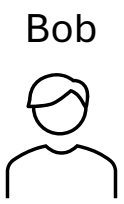
Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…



Forward Secrecy

- Recall: How the X3DH protocol computes a shared secret…



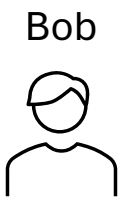
$ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B)$

1. $DH_1 = SPK_B^{ik_A}$
2. $DH_2 = IPK_B^{ek_A}$
3. $DH_3 = SPK_B^{ek_A}$
4. $DH_4 = (OPK_B)^{ek_A}$
5. $SK_A = \text{KDF}(DH_1, DH_2, DH_3, DH_4)$

Forward Secrecy

- Recall: How the X3DH protocol computes a shared secret…



$ik_B, sk_B, \{ok_B^1, \ldots, ok_B^{100}\}$

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B)$

1. $DH_1 = SPK_B^{ik_A}$
2. $DH_2 = IPK_B^{ek_A}$
3. $DH_3 = SPK_B^{ek_A}$
4. $DH_4 = (OPK_B)^{ek_A}$
5. $SK_A = \text{KDF}(DH_1, DH_2, DH_3, DH_4)$

ek_A is not a long-term secret

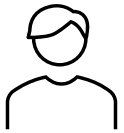
Forward Secrecy

- Recall: How the X3DH protocol computes a shared secret…

Alice

Leakage!

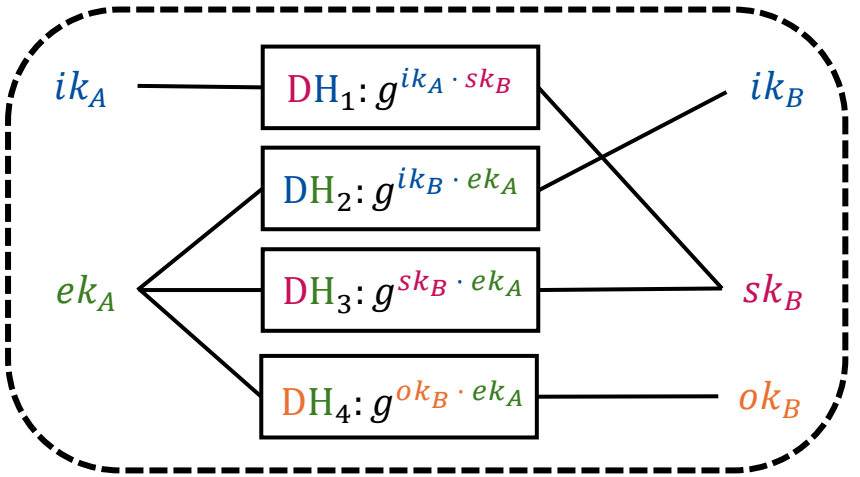
Bob



$ik_B, sk_B, \{ok_B^1, \ldots, ok_B^{100}\}$

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B)$

1. $\text{DH}_1 = SPK_B^{ik_A}$
2. $\text{DH}_2 = IPK_B^{ek_A}$
3. $\text{DH}_3 = SPK_B^{ek_A}$
4. $\text{DH}_4 = (OPK_B)^{ek_A}$
5. $SK_A = \text{KDF}(\text{DH}_1, \text{DH}_2, \text{DH}_3, \text{DH}_4)$



Forward Secrecy

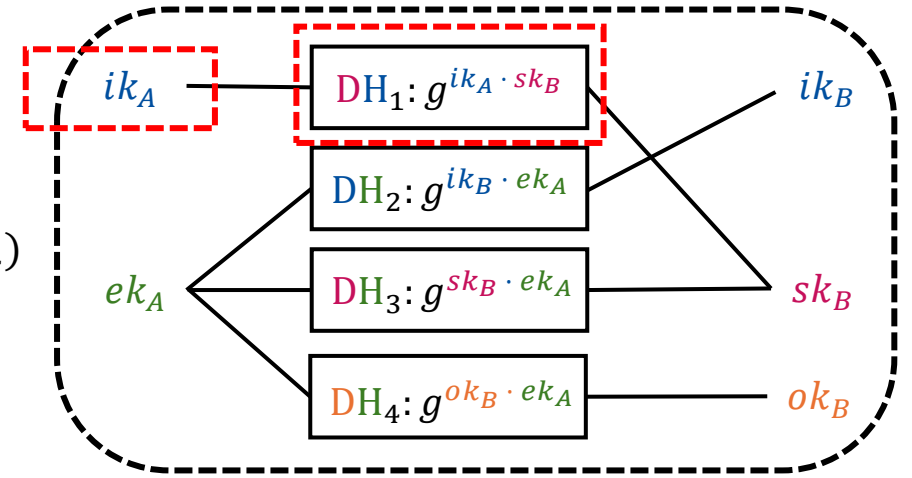
- Recall: How the X3DH protocol computes a shared secret…

Alice

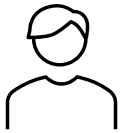
Leakage!

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B)$

1. $\text{DH}_1 = SPK_B^{ik_A}$
2. $\text{DH}_2 = IPK_B^{ek_A}$
3. $\text{DH}_3 = SPK_B^{ek_A}$
4. $\text{DH}_4 = (OPK_B)^{ek_A}$
5. $SK_A = \text{KDF}(\text{DH}_1, \text{DH}_2, \text{DH}_3, \text{DH}_4)$



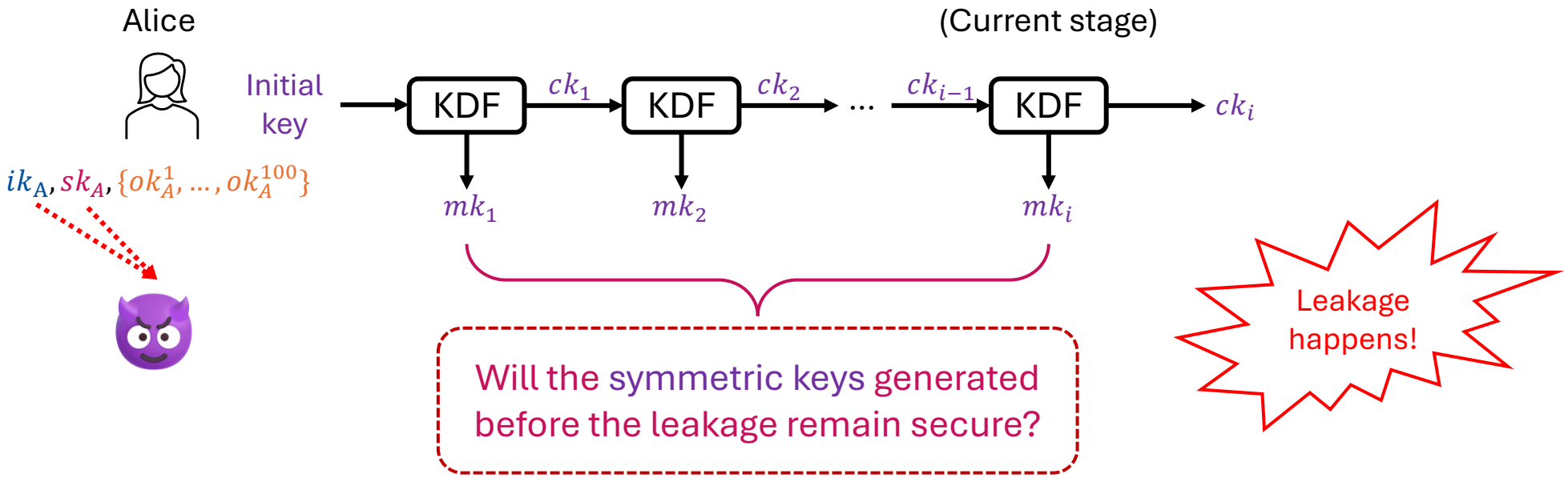
Bob



$ik_B, sk_B, \{ok_B^1, \ldots, ok_B^{100}\}$

Forward Secrecy

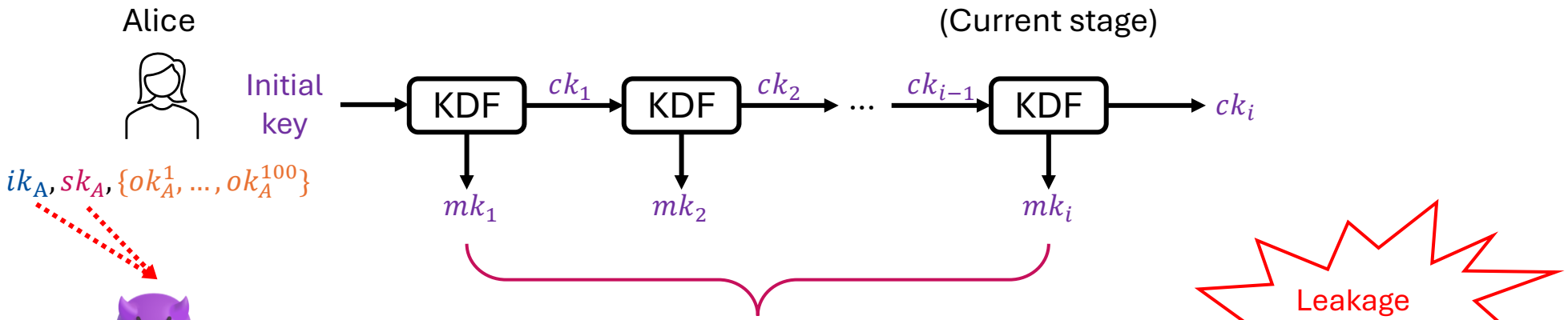
- Long-term secret keys are compromised, but past communication remains secure…



Initial_key (of X3DH)= $\mathrm{KDF}(\mathrm{DH}_1, \mathrm{DH}_2, \mathrm{DH}_3, \mathrm{DH}_4)$

Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…



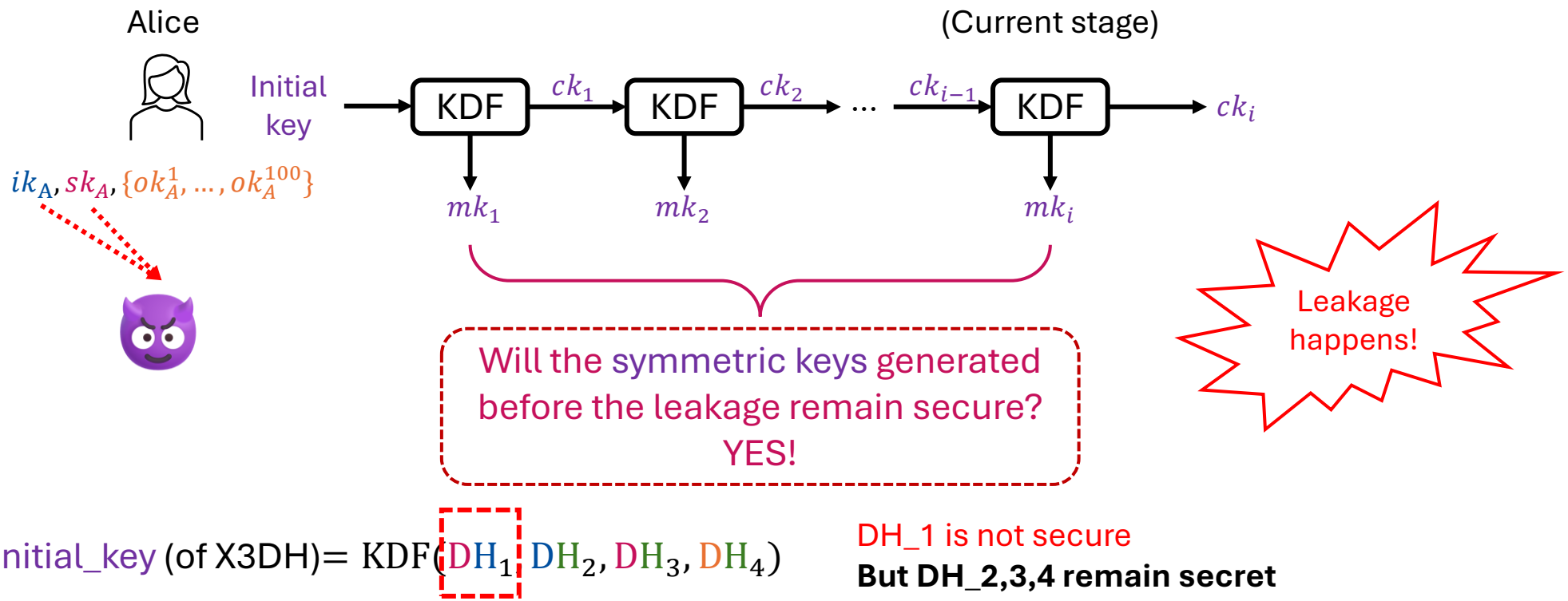
Initial_key (of X3DH)= $\mathrm{KDF}(\mathrm{DH}_1, \mathrm{DH}_2, \mathrm{DH}_3, \mathrm{DH}_4)$

DH_1 is not secure

But DH_2,3,4 remain secret

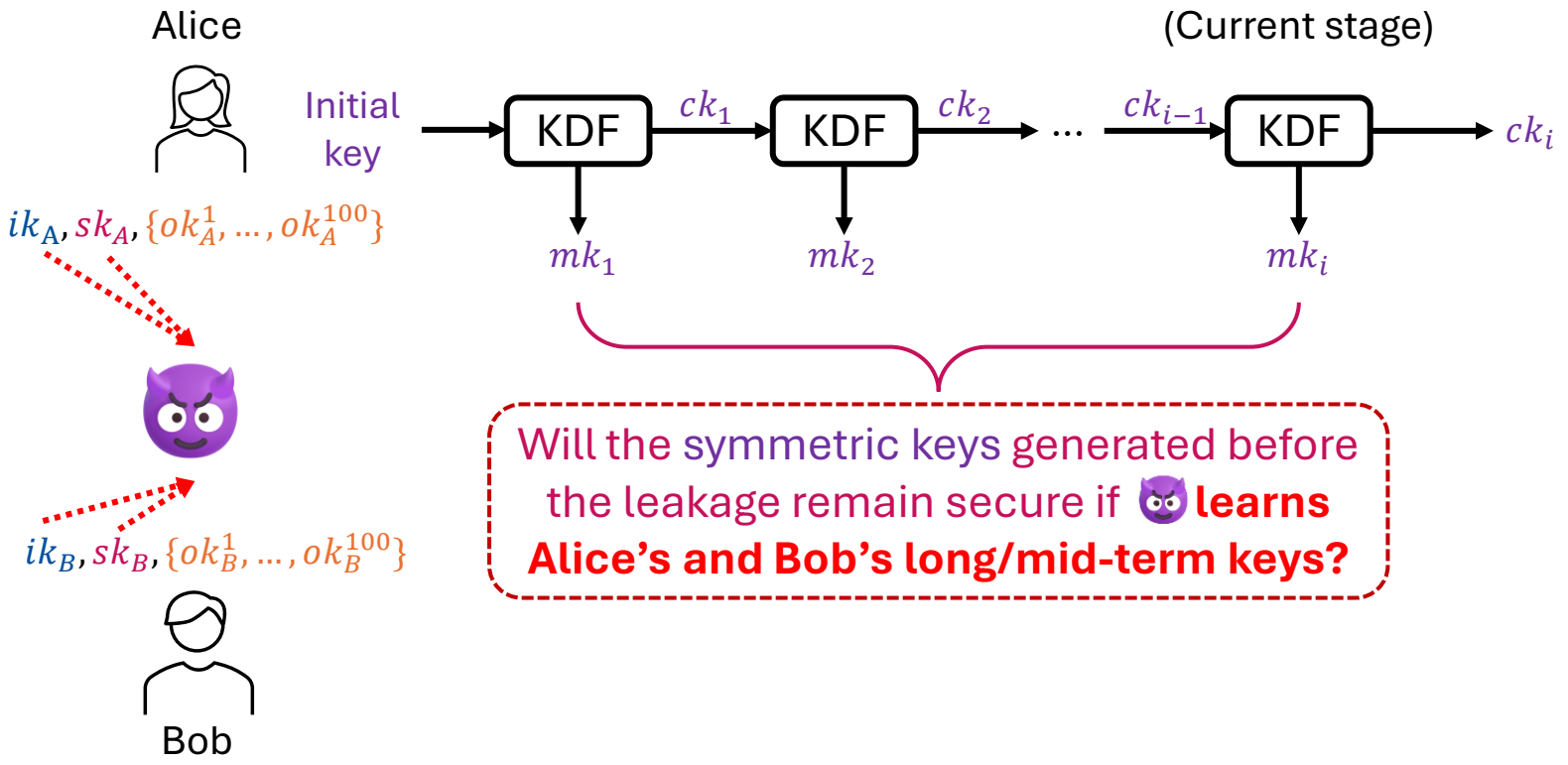
Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…



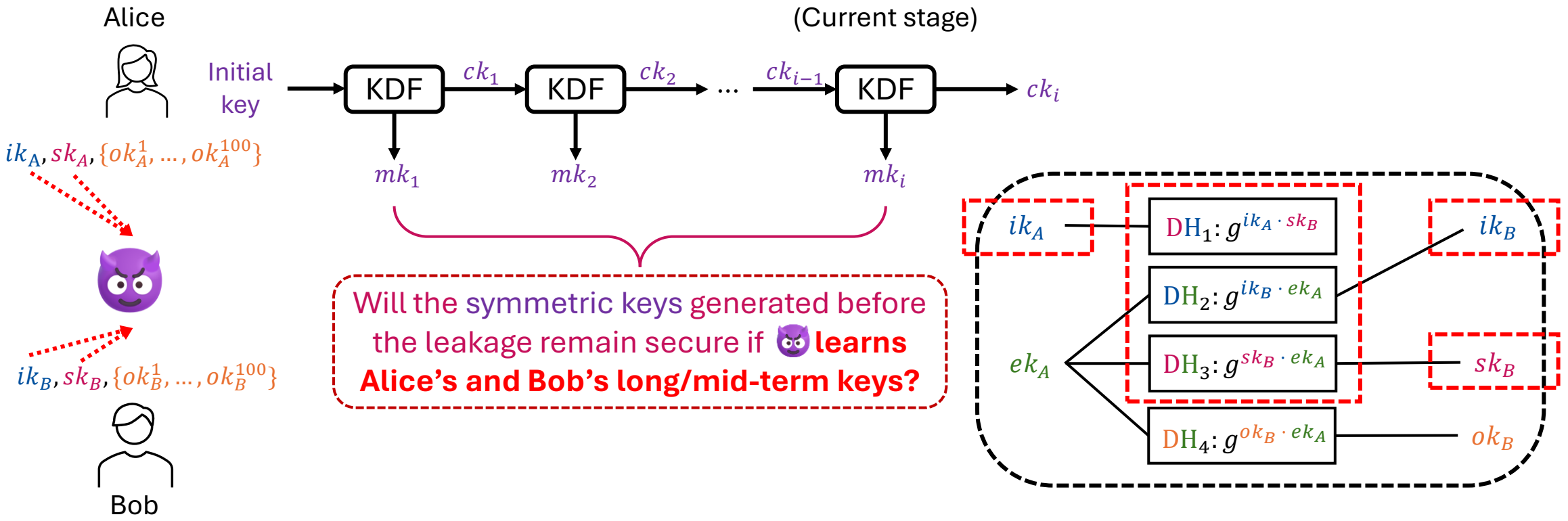
Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…

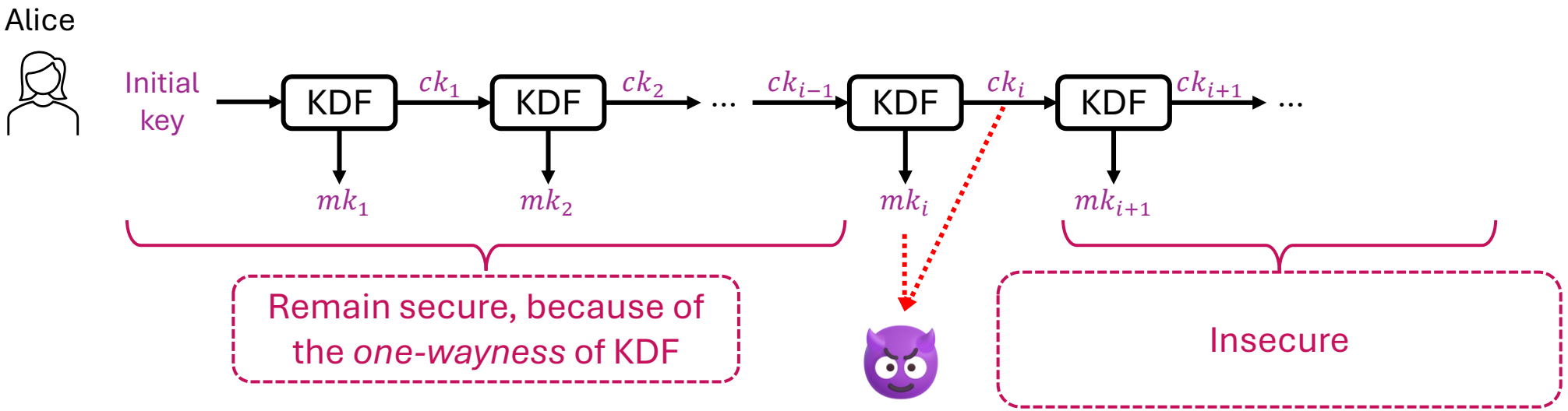


Forward Secrecy

- Long-term secret keys are compromised, but past communication remains secure…

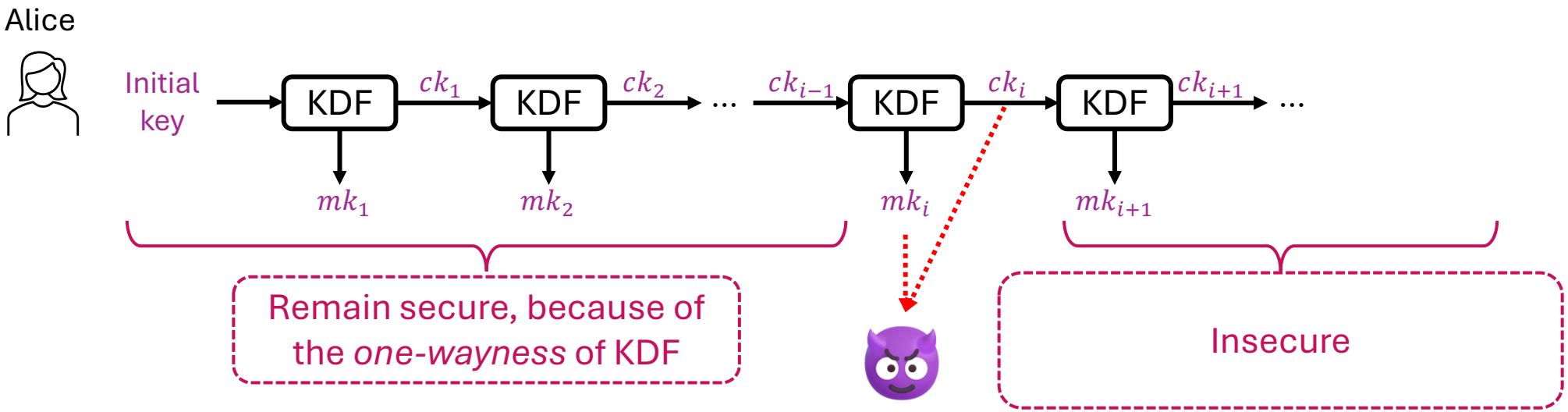


Backward Secrecy

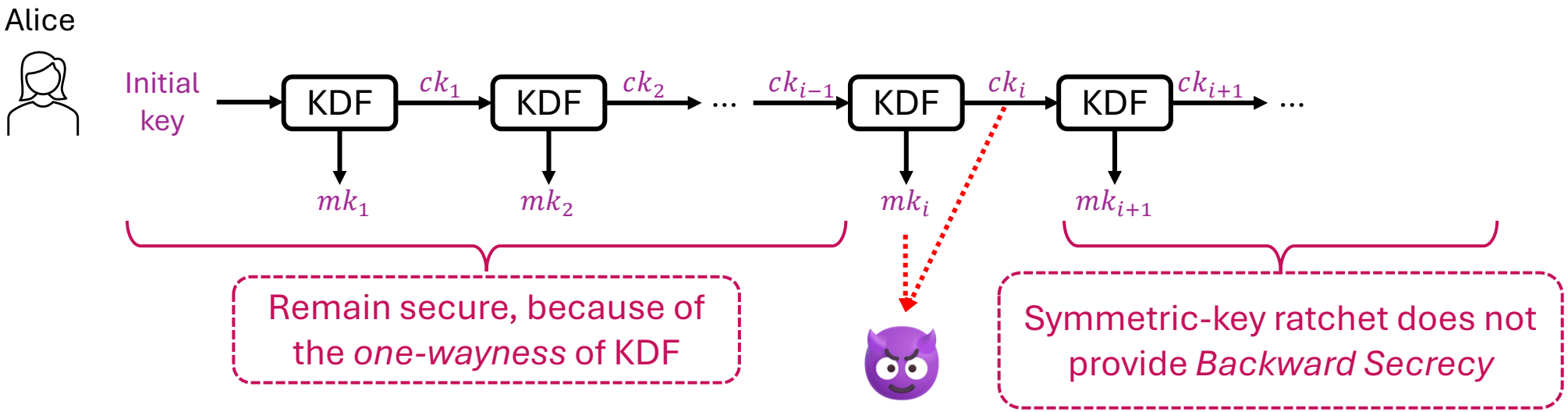


Backward Secrecy

- Backward Secrecy: Future communication remains secure even if a current session key is compromised



Backward Secrecy



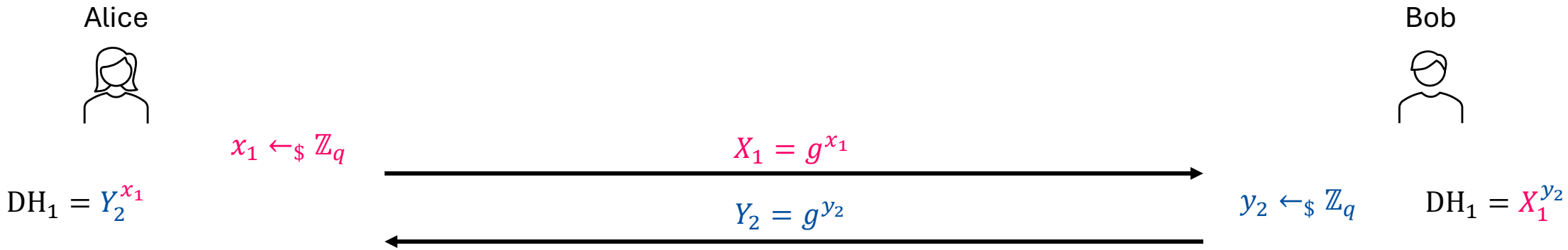
Diffie-Hellman Ratchet

- X3DH + Symmetric-key Ratchet
  - X3DH provides *Forward Secrecy*
  - Current session key compromises does not lead to the compromise of previous session keys
    - (by the one-wayness of KDF in Symmetric-key Ratchet)
  - No Backward Secrecy

- Solution: Diffie-Hellman Ratchet (Today)

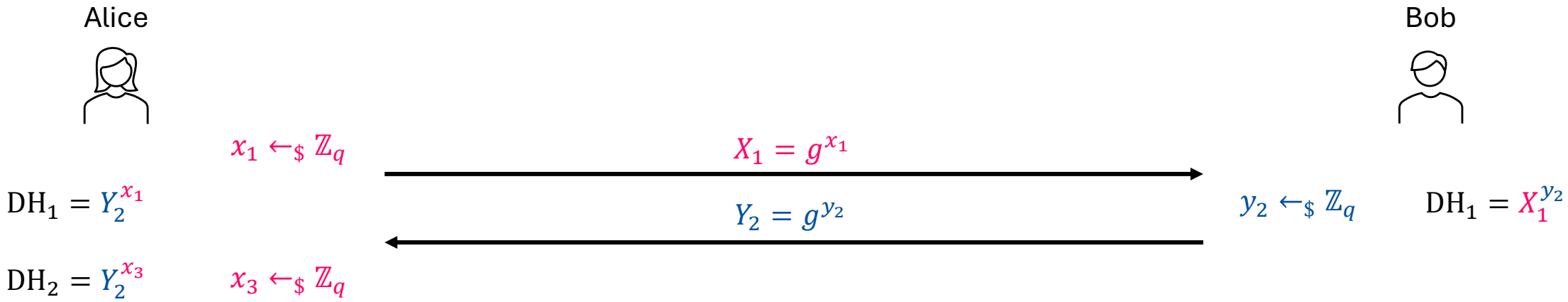
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*…



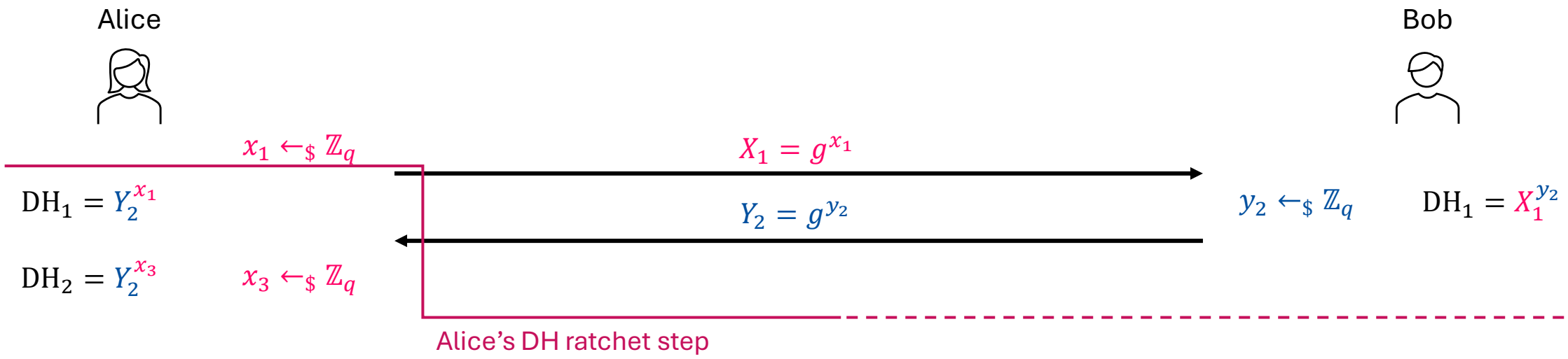
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



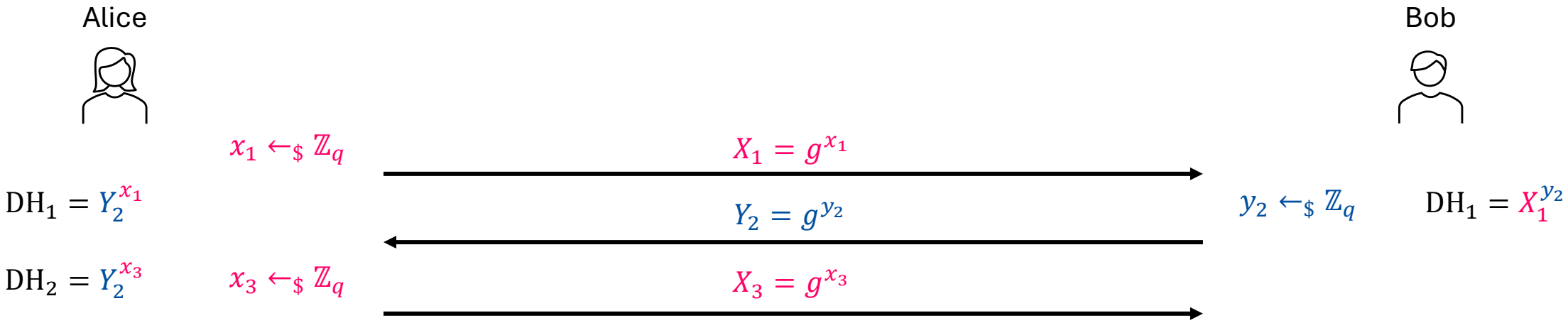
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



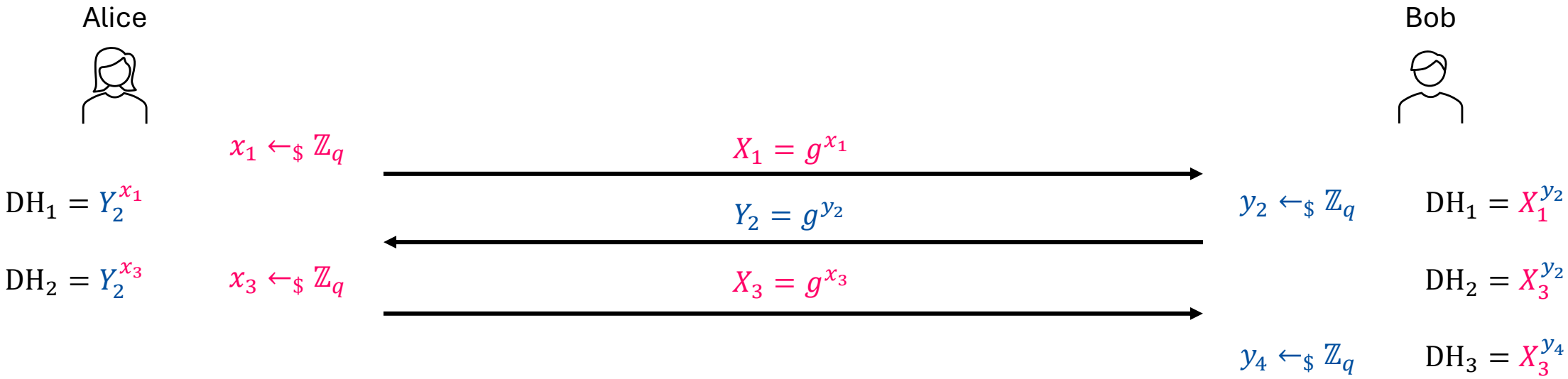
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



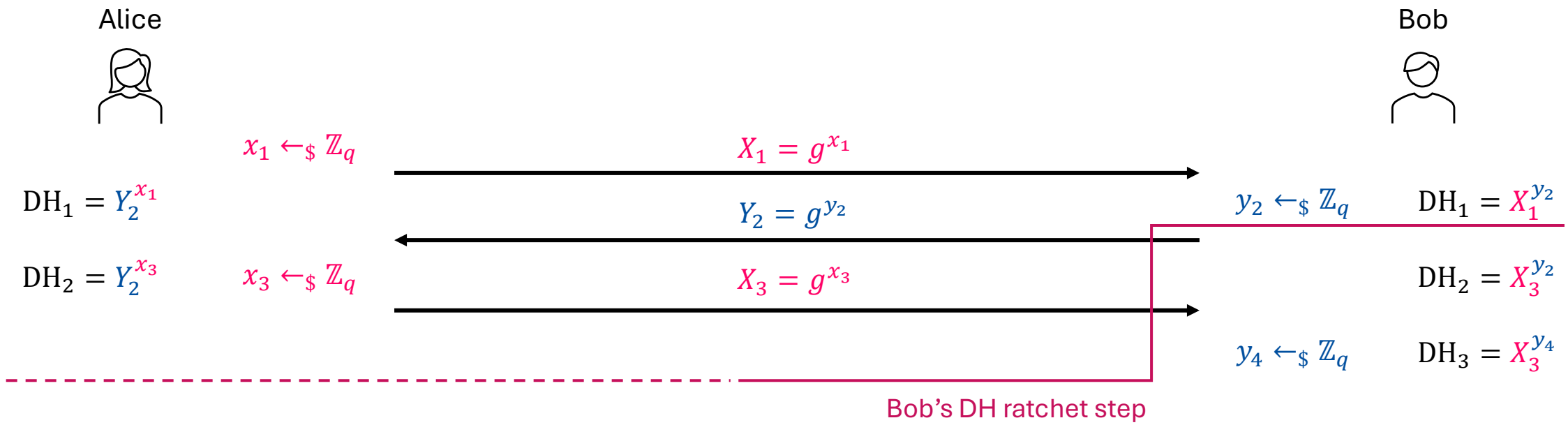
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



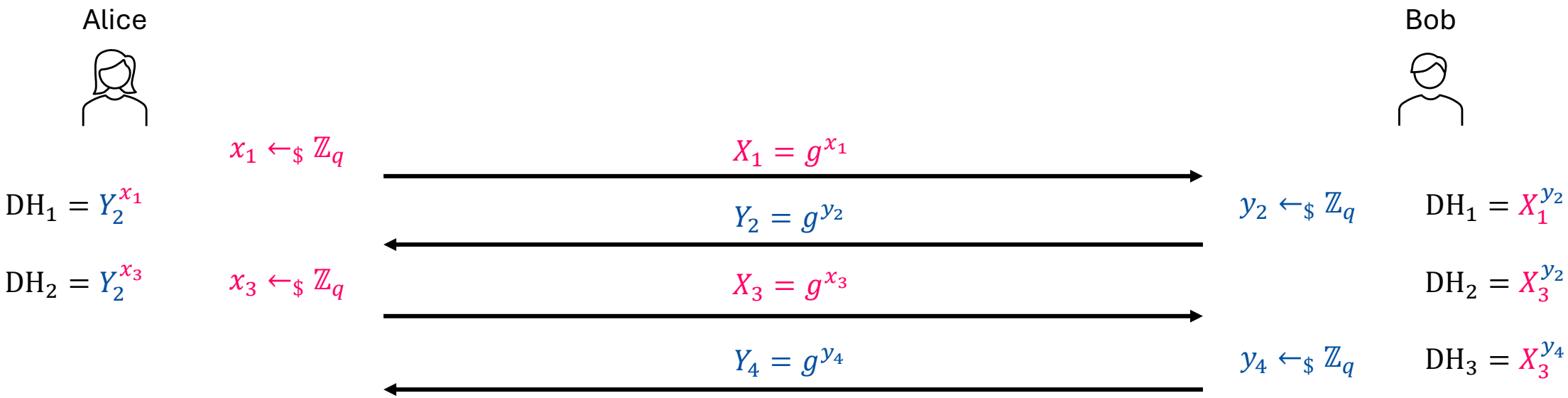
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*…



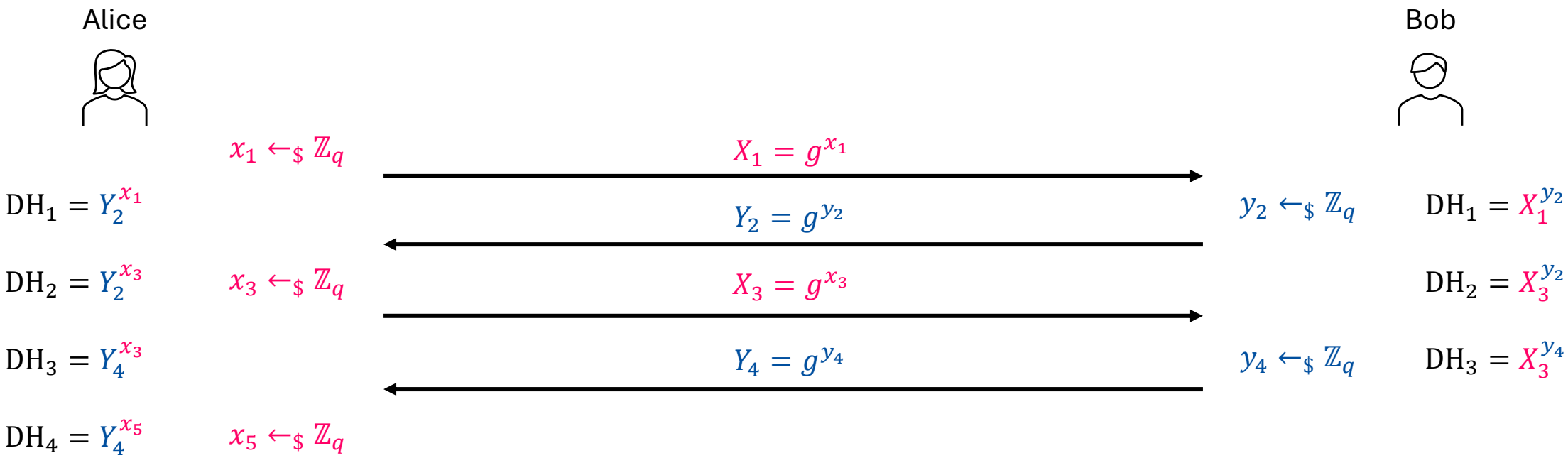
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



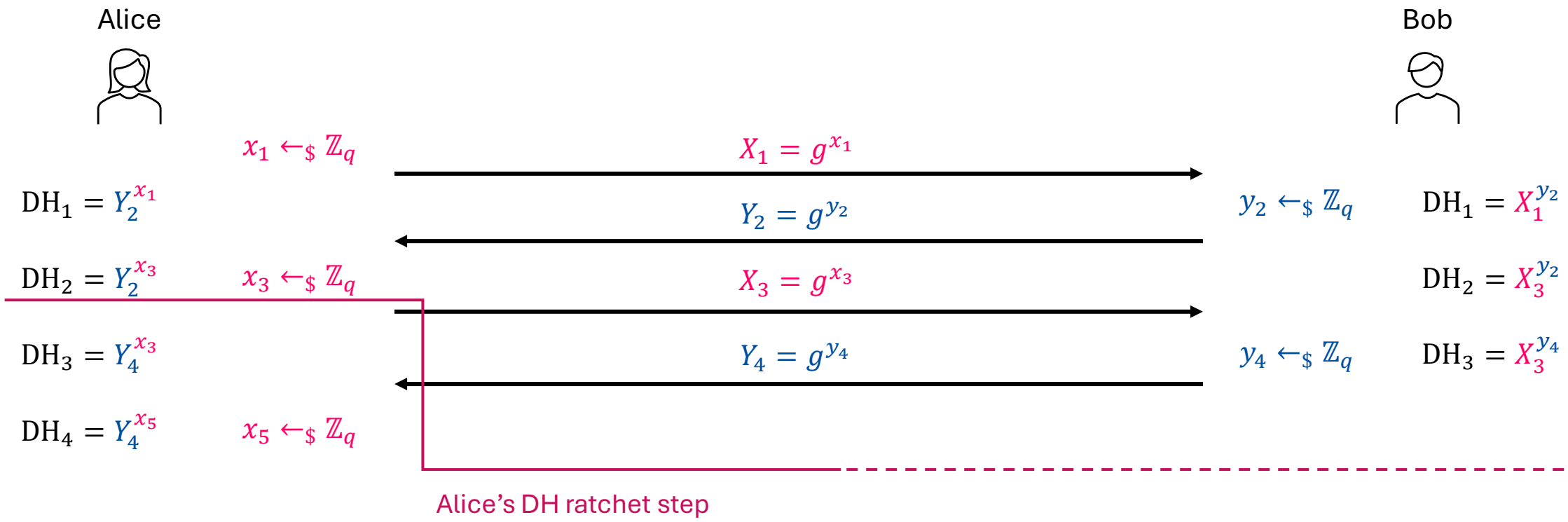
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



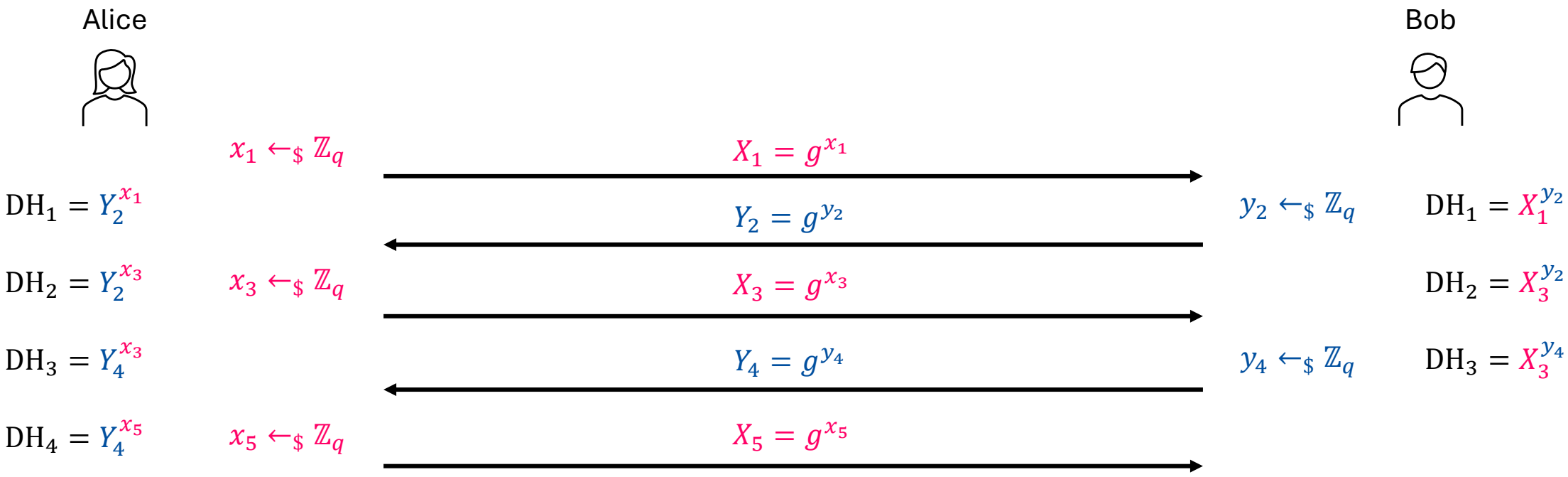
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



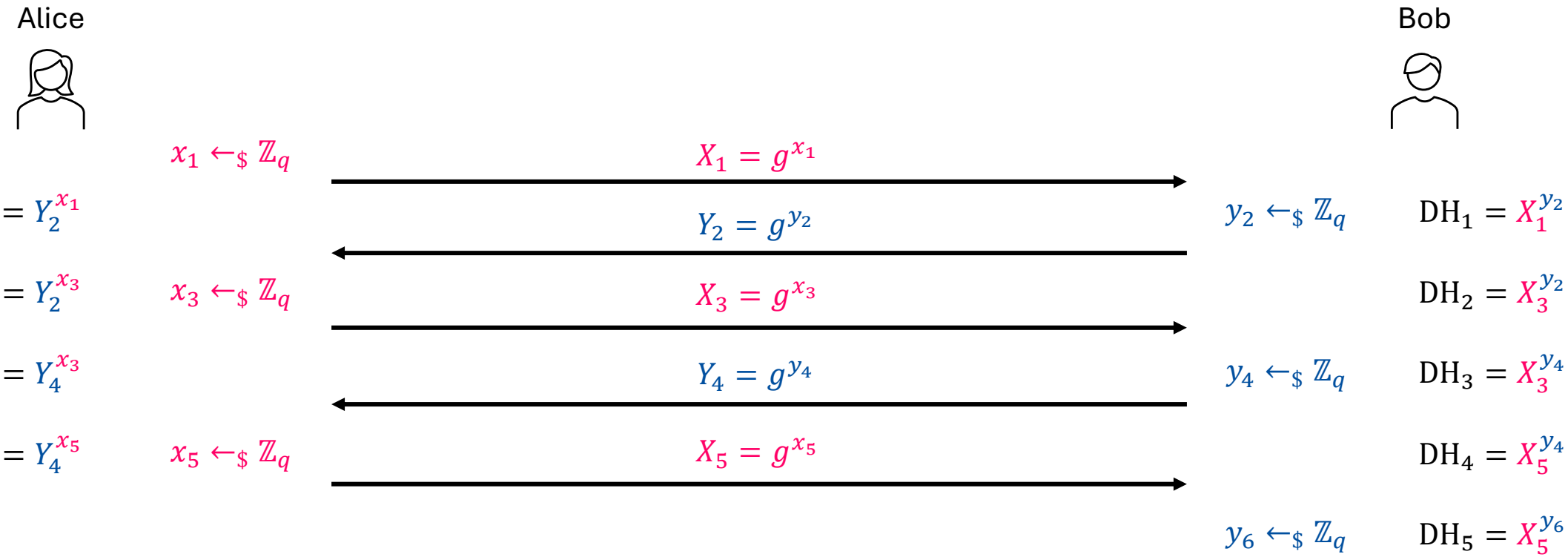
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



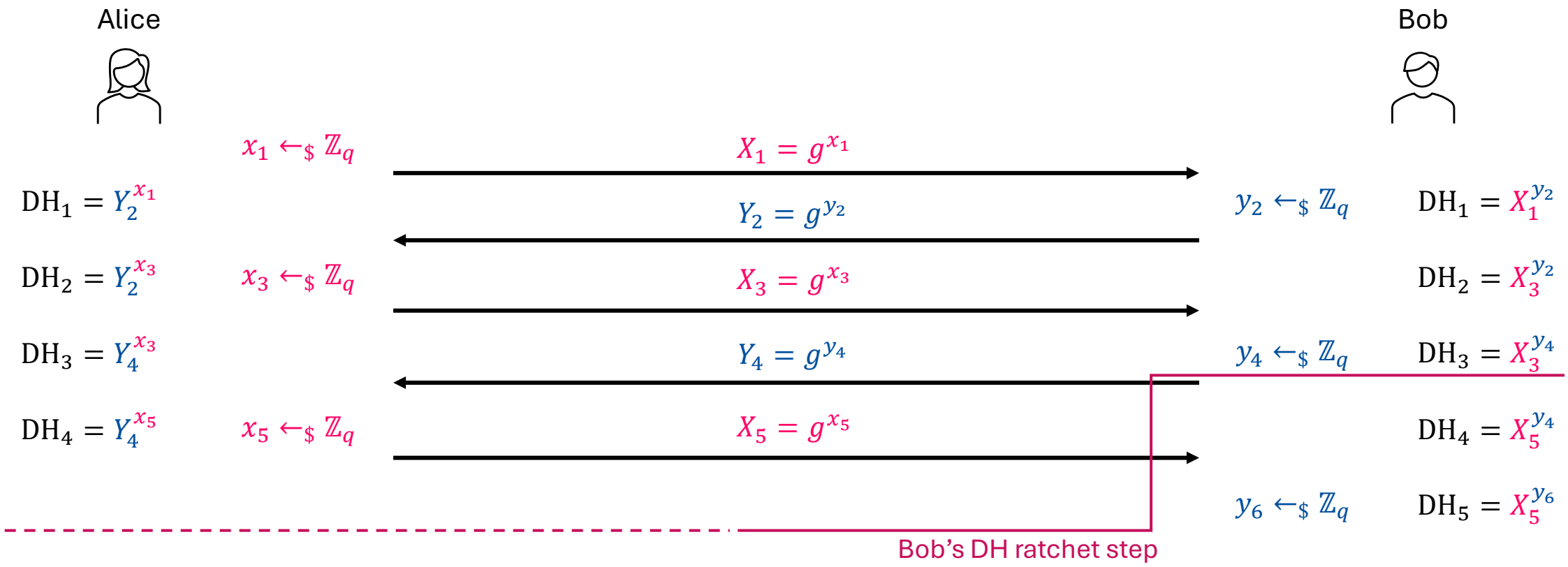
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*...



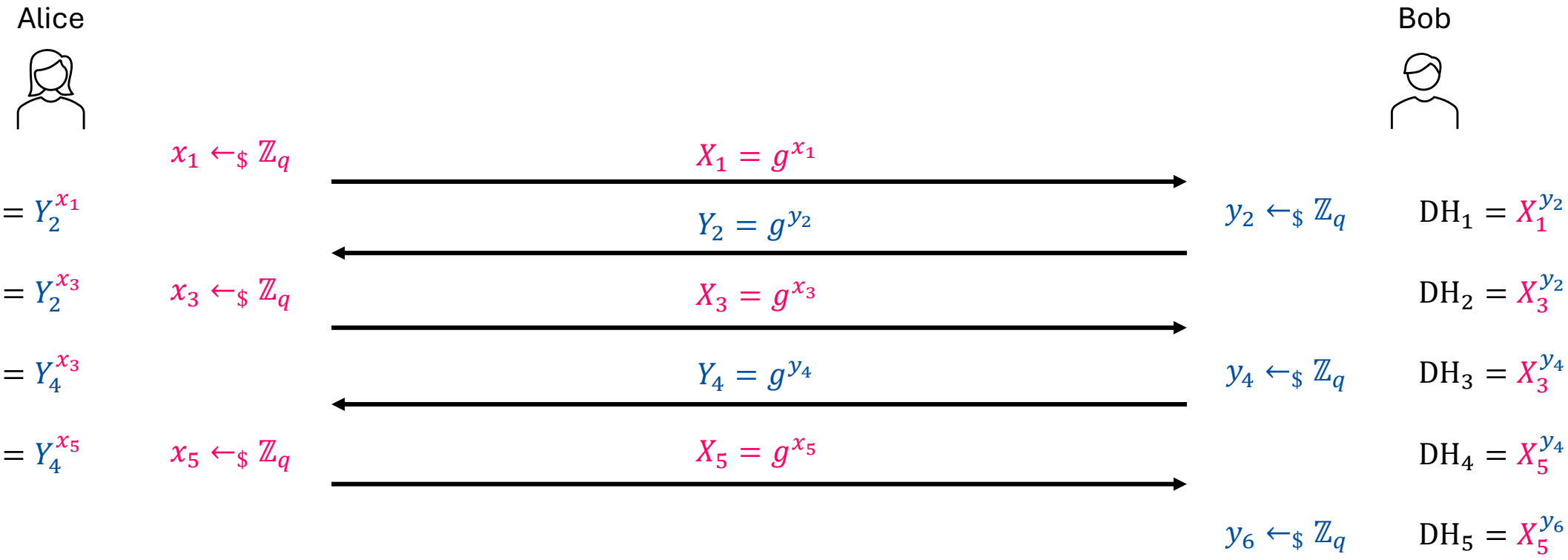
Diffie-Hellman Ratchet

- A toy example: Running DHKE continuously with *rotating ephemeral keys*…



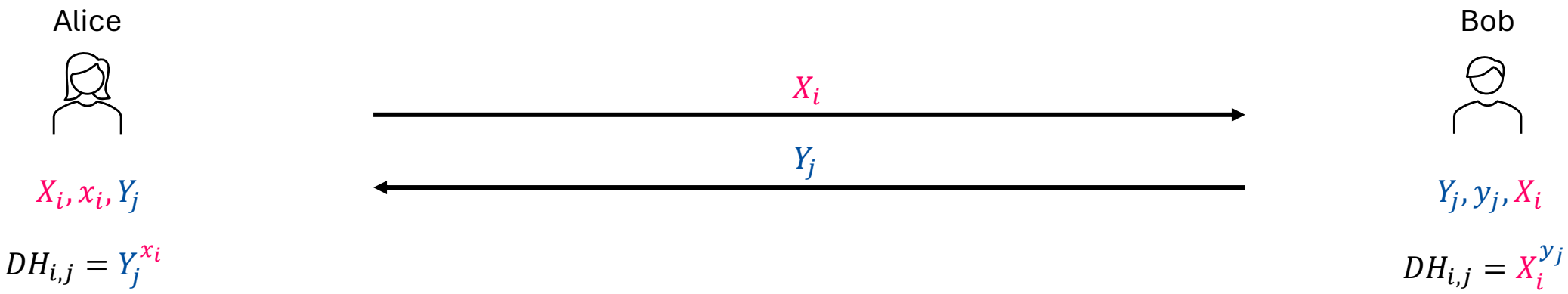
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



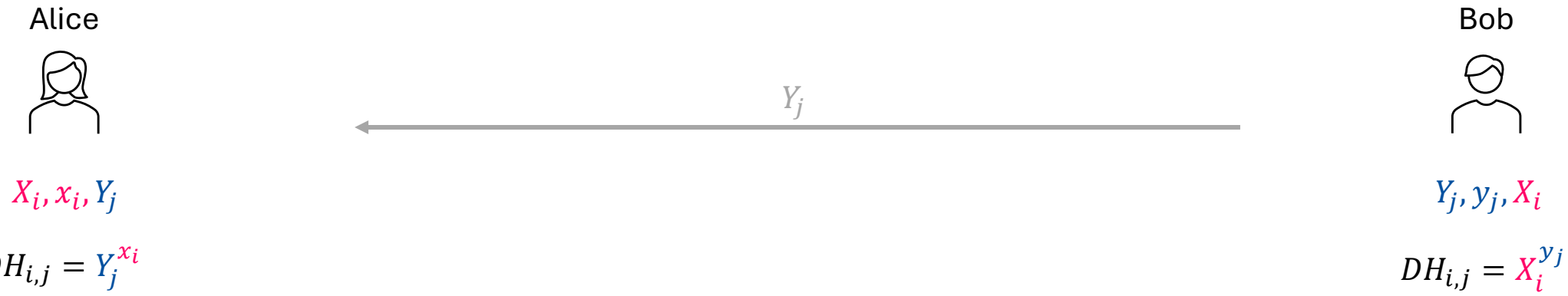
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



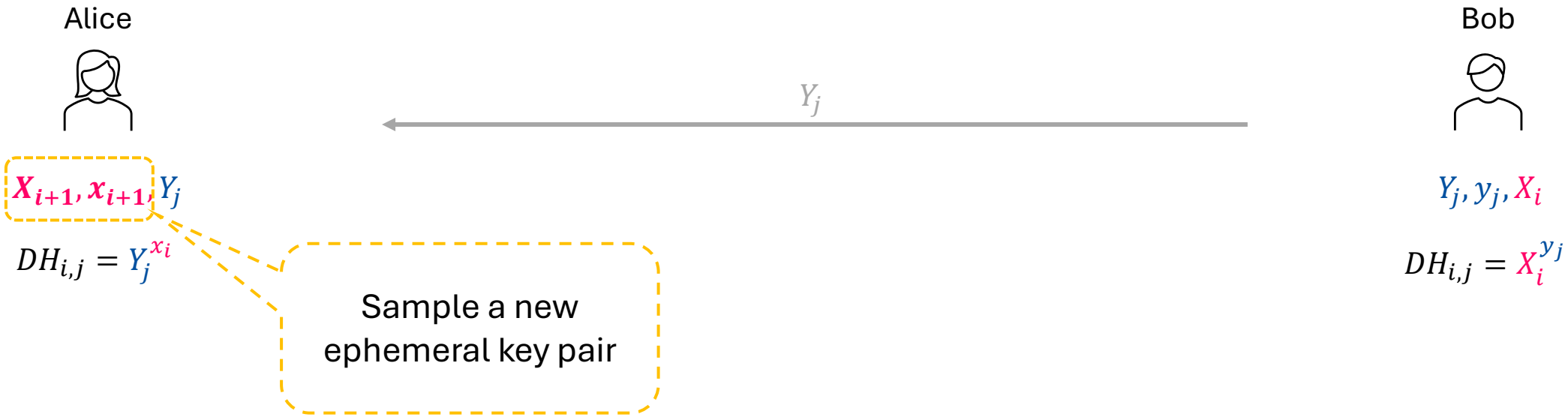
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



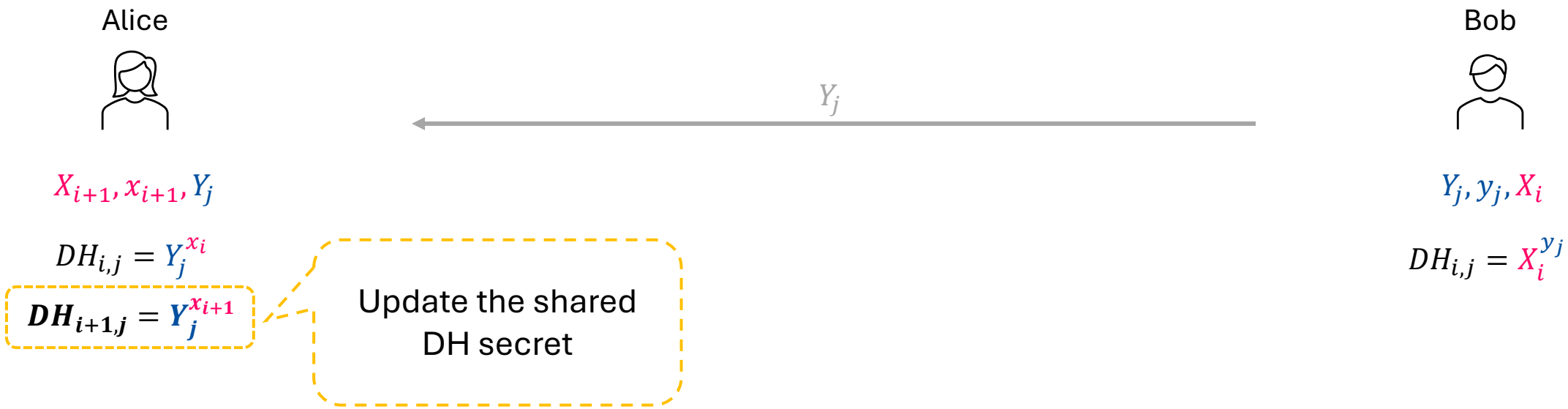
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



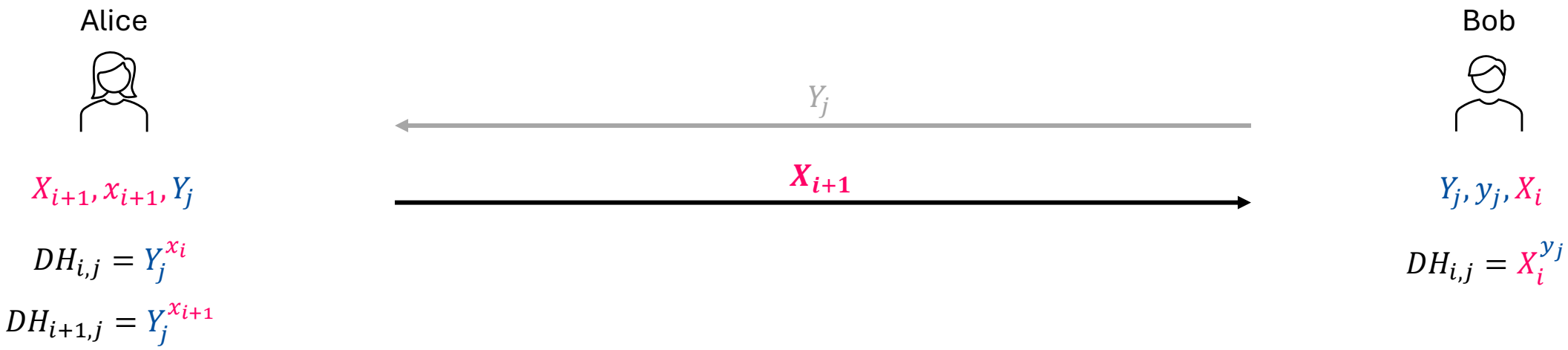
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



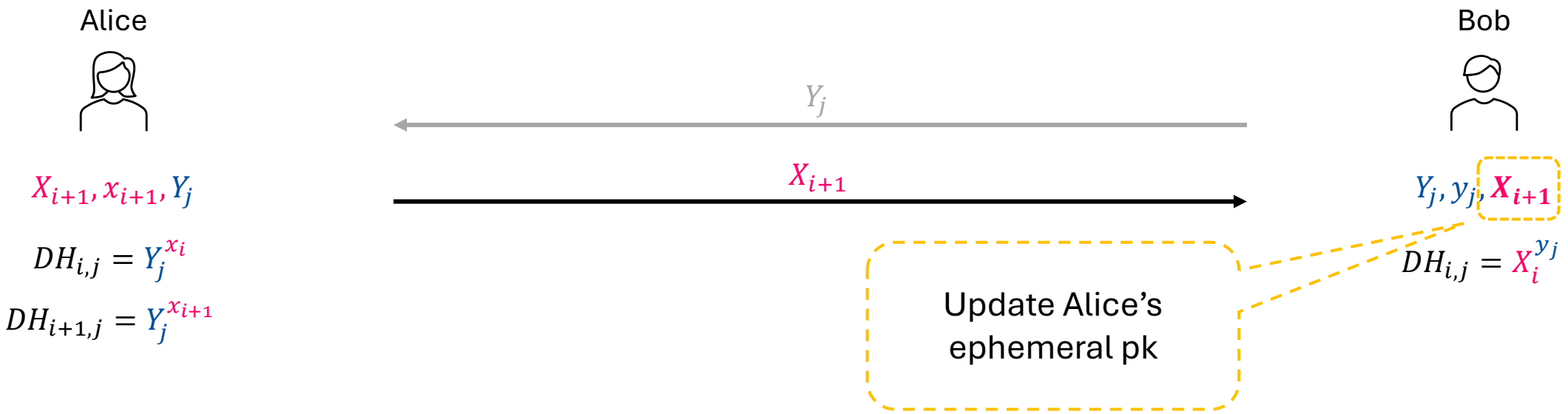
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



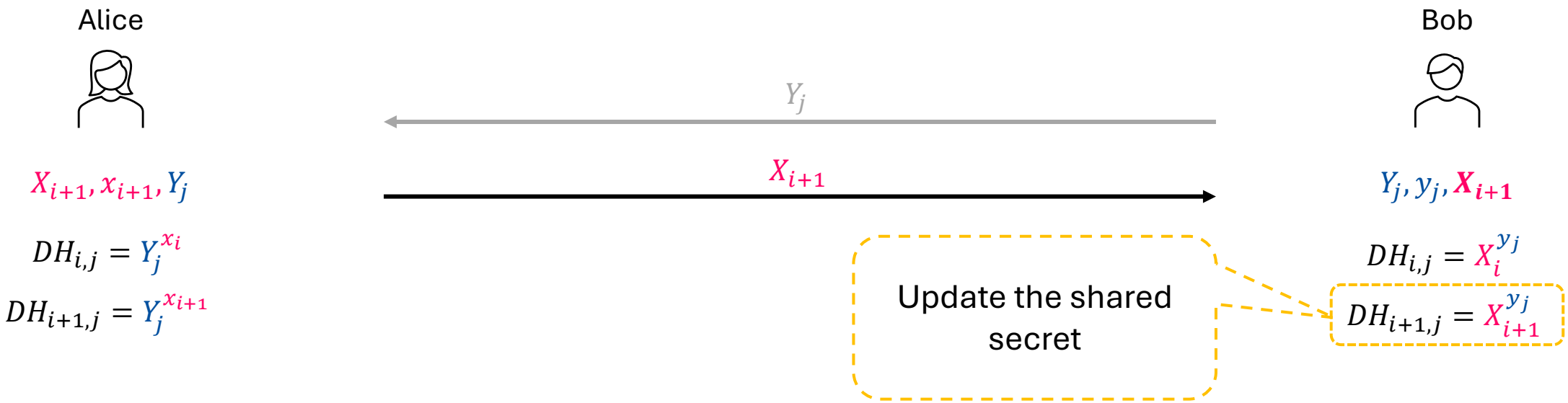
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



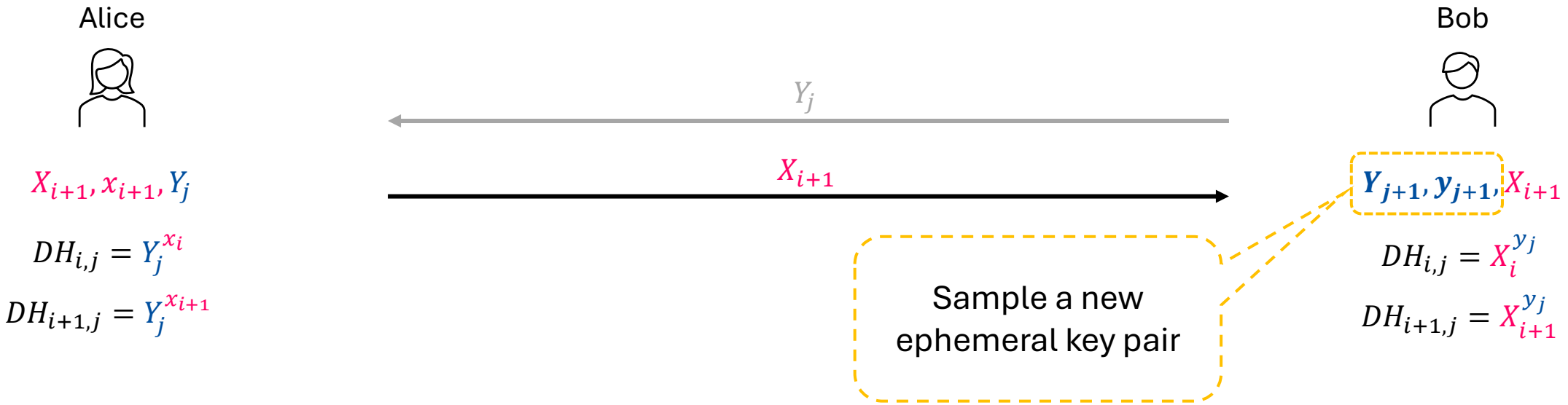
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



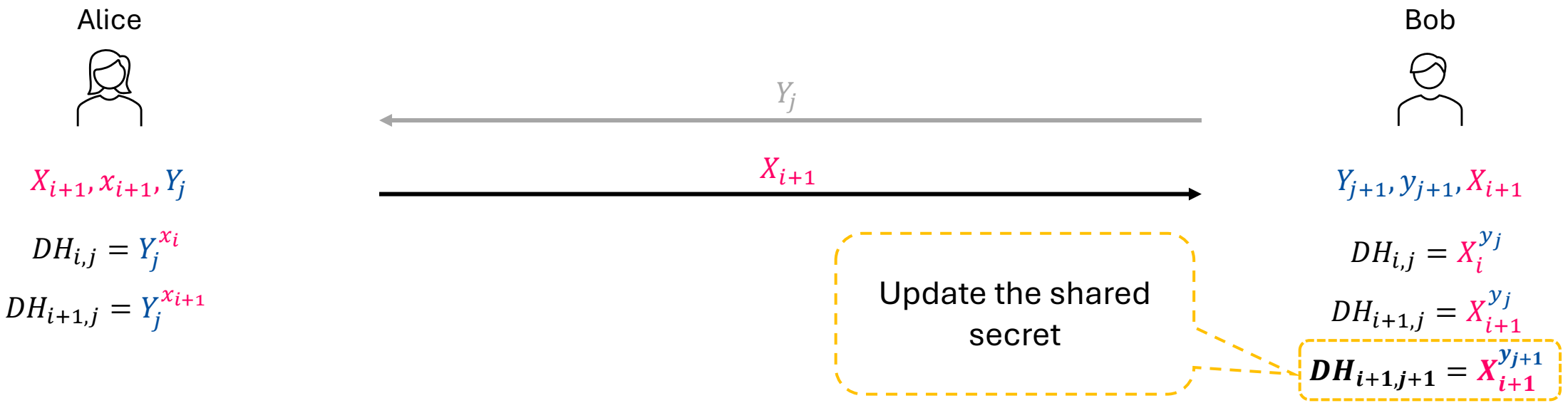
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



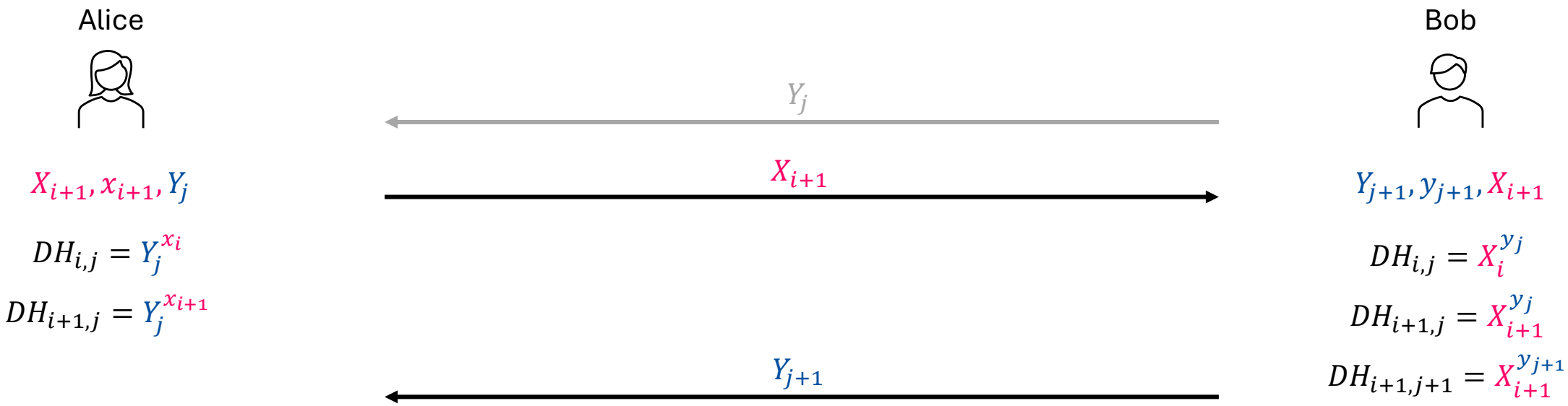
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



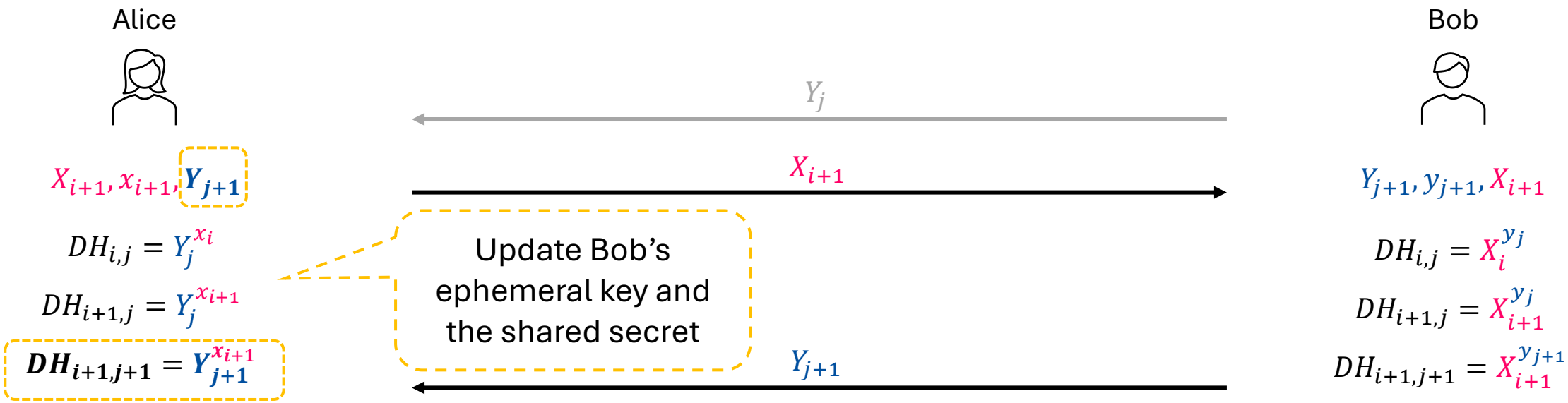
Double Ratchet – DH Ratchet

- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



Double Ratchet – DH Ratchet

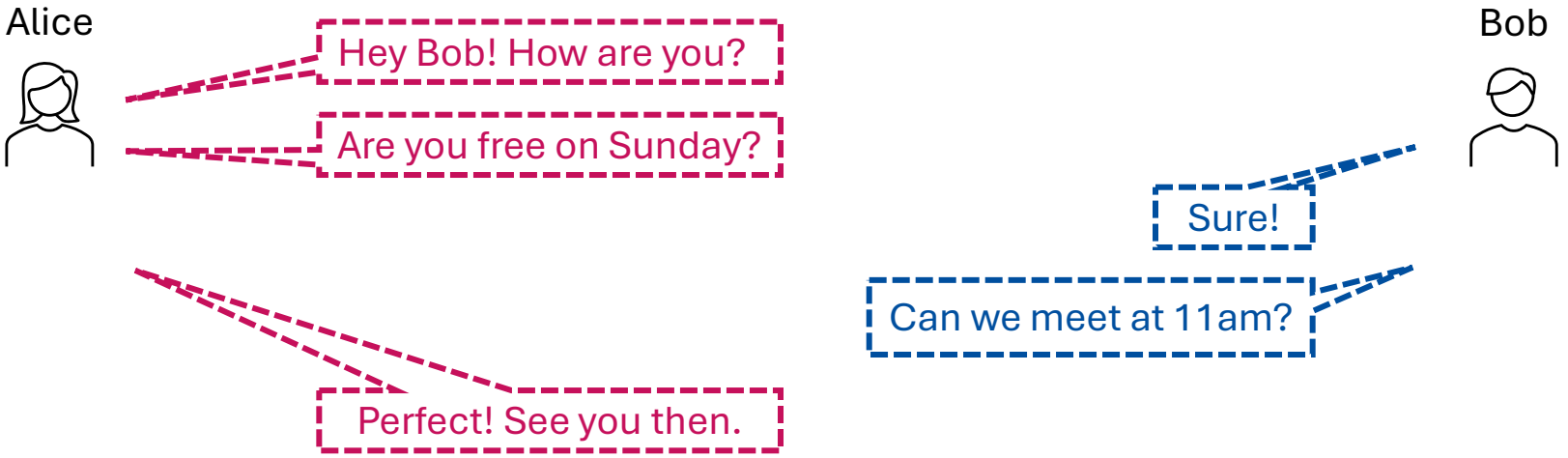
- Main idea of DH Ratchet: Running DHKE continuously with *rotating ephemeral keys*…



Double Ratchet

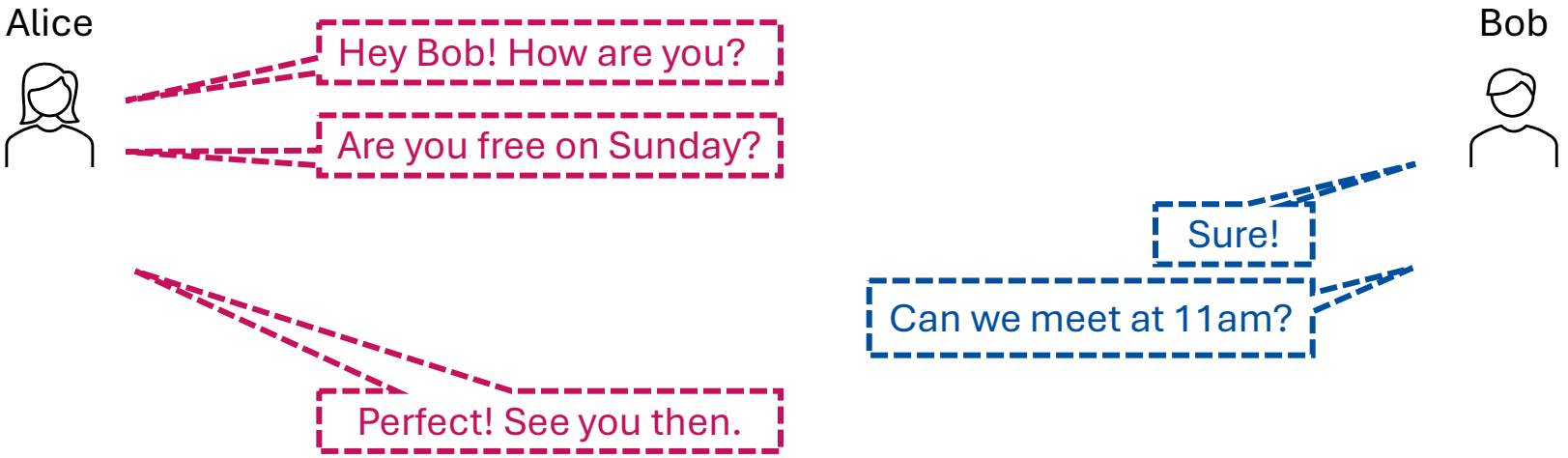
- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet
  - When a party sends messages (**before** its peer party replies): Use Symmetric-key Ratchet…
  - When the peer party replies: Use Diffie-Hellman Ratchet to update the key…

- Example:



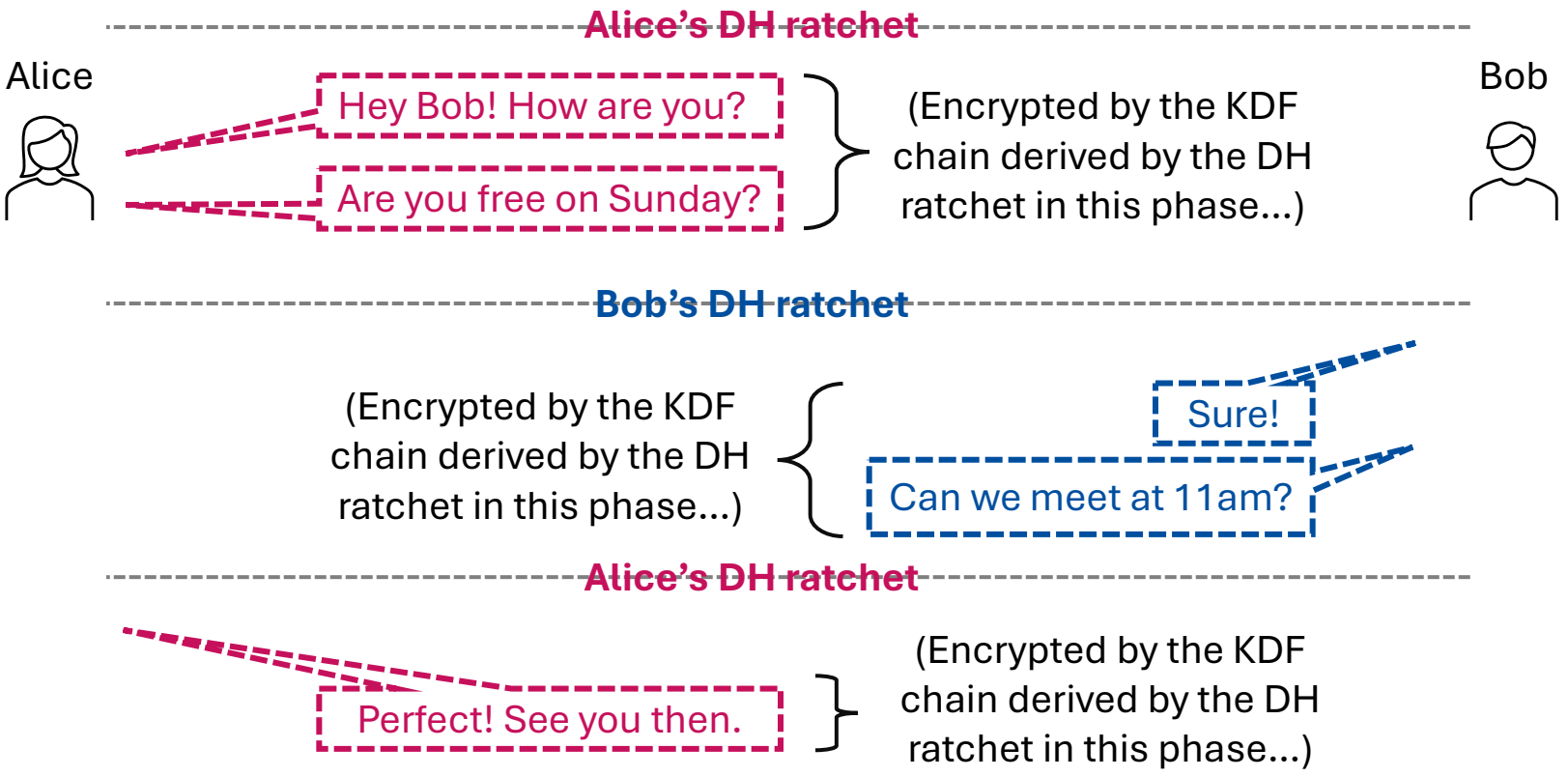
Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet



Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet

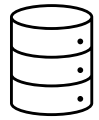


Double Ratchet

Alice

X_i, x_i, Y_j

Root key
(from previous stage)



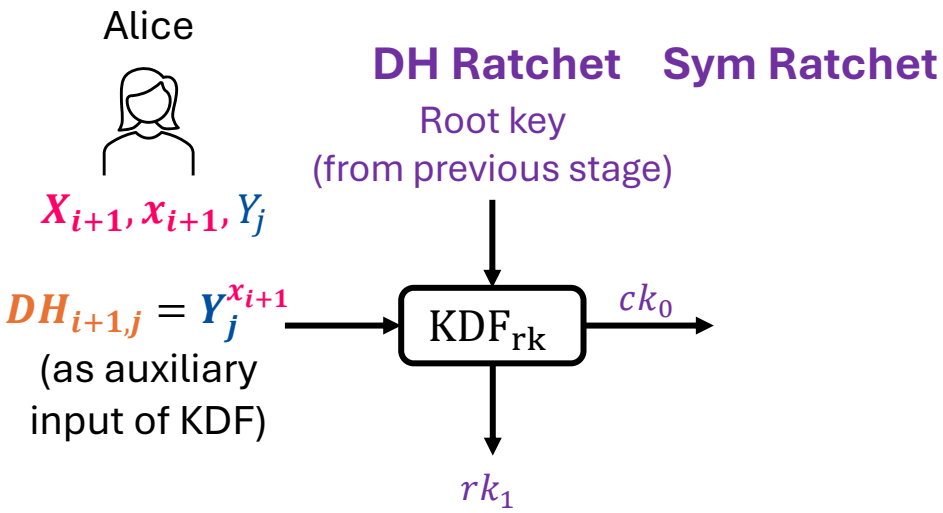
All messages
are relayed by
the server

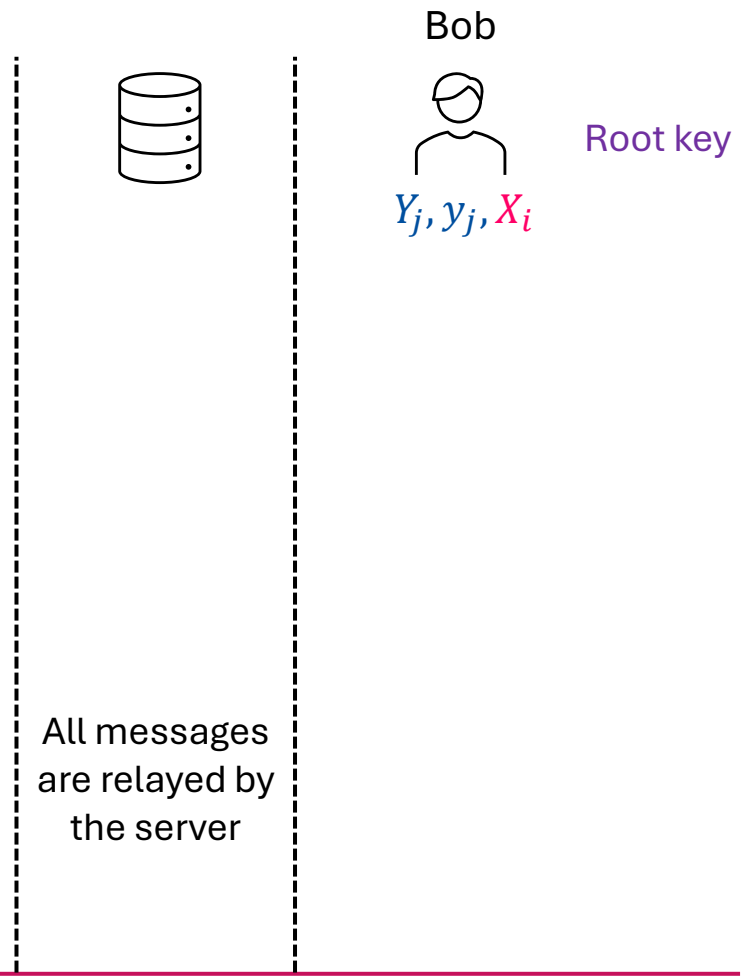
Bob

Y_j, y_j, X_i

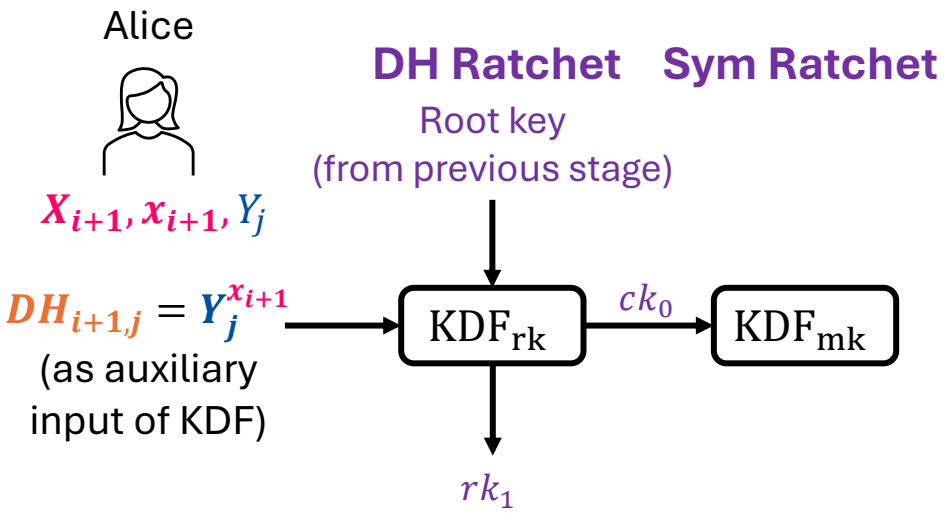
Root key

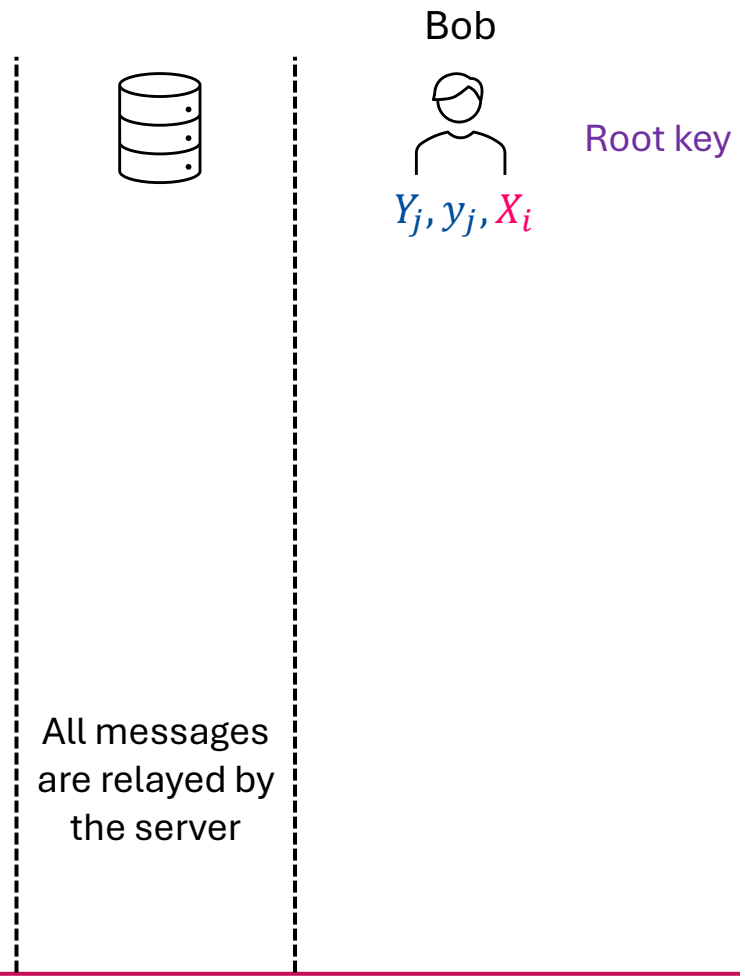
Double Ratchet



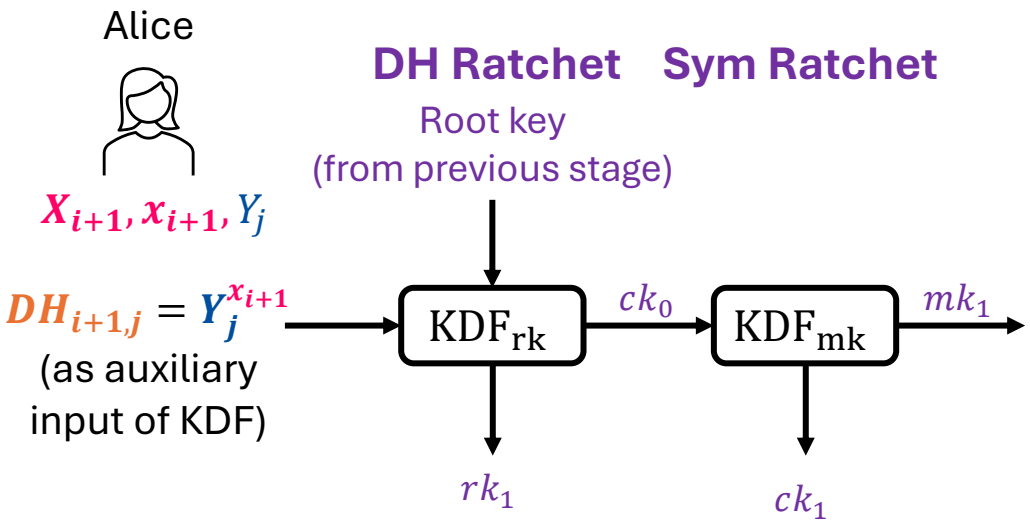


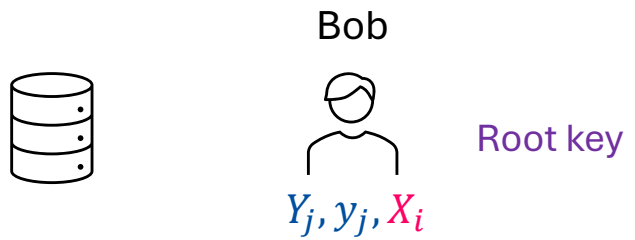
Double Ratchet



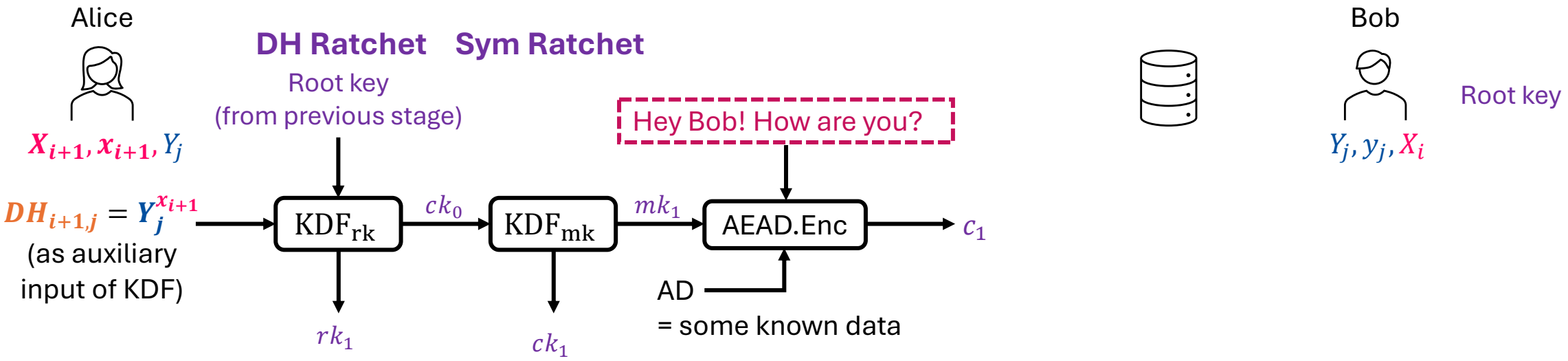


Double Ratchet

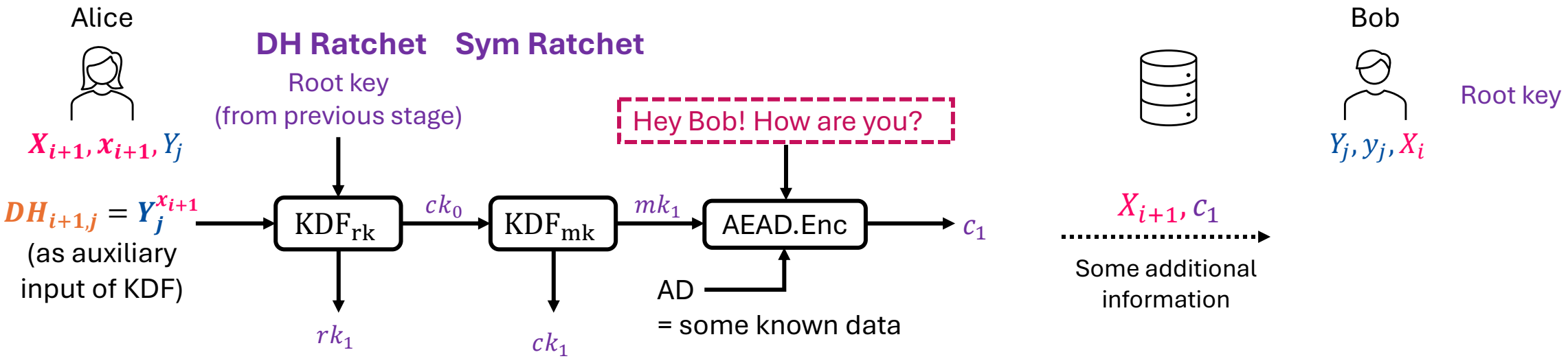




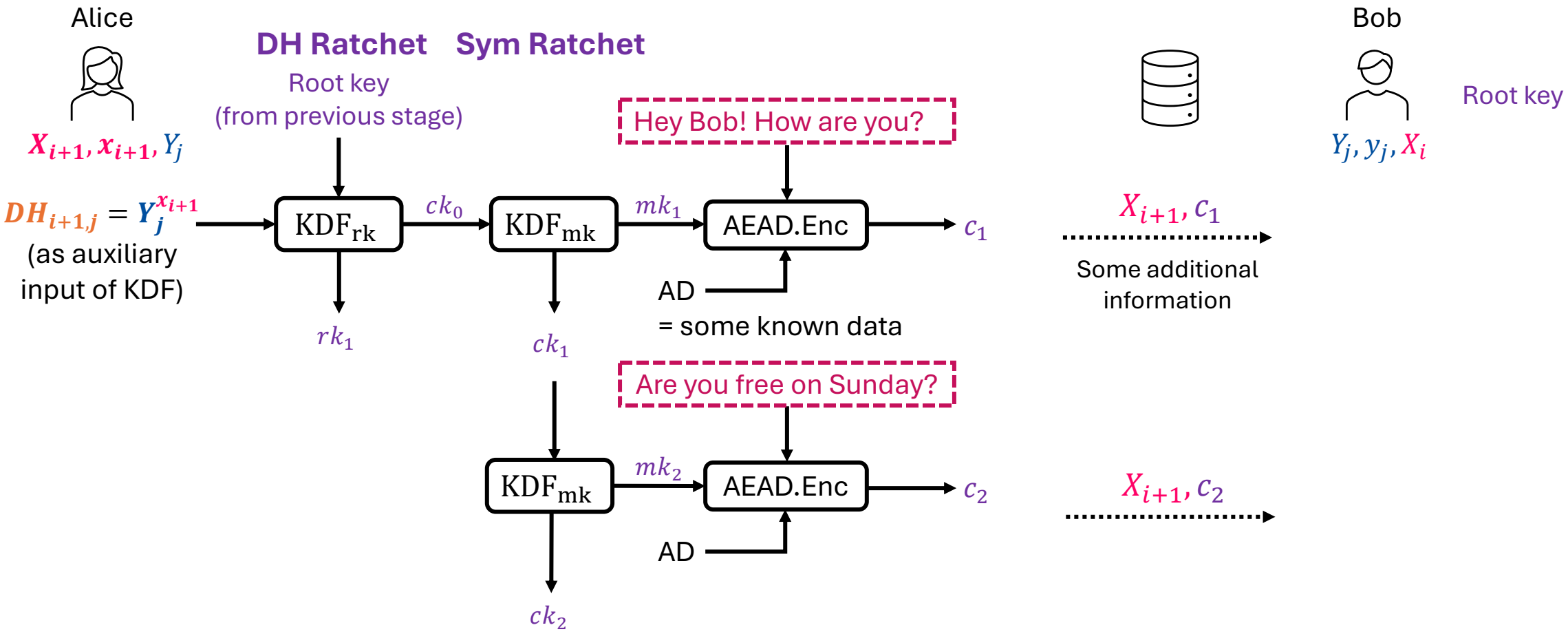
Double Ratchet



Double Ratchet



Double Ratchet

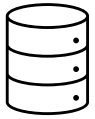


Double Ratchet

Alice

X_{i+1}, x_{i+1}, Y_j

$DH_{i+1,j} = Y_j^{x_{i+1}}$



Root key

Bob

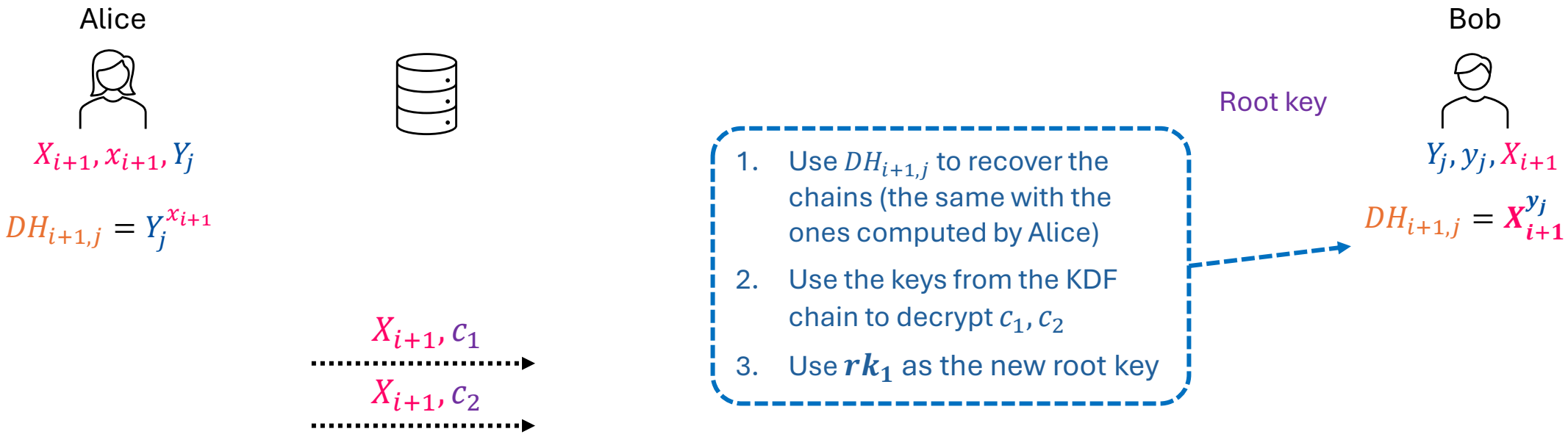
Y_j, y_j, X_i

X_{i+1}, c_1

X_{i+1}, c_2

All messages are relayed by the server

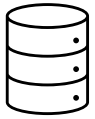
Double Ratchet



Double Ratchet

Alice

X_{i+1}, x_{i+1}, Y_j



New root key
$(= \boldsymbol{rk_1})$

Bob

$\boldsymbol{Y_{j+1}}, \boldsymbol{y_{j+1}}, X_{i+1}$

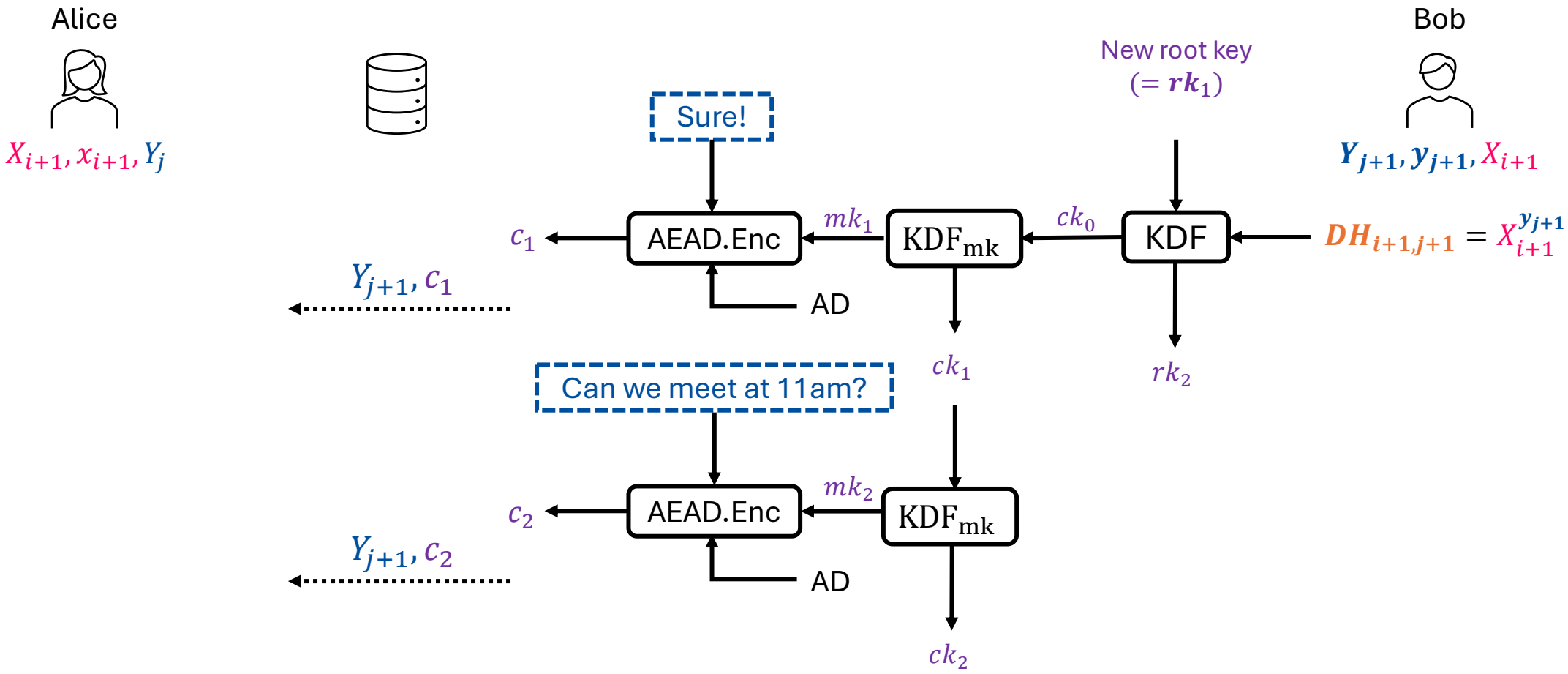
After decrypting Alice's messages, Bob

(1) Use the $\boldsymbol{rk_1}$ as the new root key

(2) Sample new ephemeral key pair

$\boldsymbol{DH_{i+1,j+1}} = X_{i+1}^{\boldsymbol{y_{j+1}}}$

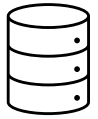
Double Ratchet



Double Ratchet

Alice

$X_{i+1}, x_{i+1}, Y_{j+1}$



Bob

$Y_{j+1}, y_{j+1}, X_{i+1}$

Y_{j+1}, c_1

Y_{j+1}, c_2

$DH_{i+1,j+1} = Y_{j+1}^{x_{i+1}}$

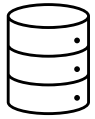
Use $DH_{i+1,j+1}$ to recover two chains to decrypt c_1, c_2.

Double Ratchet

Alice

$X_{i+2}, x_{i+2}, Y_{j+1}$

New root key
$(= rk_2)$



Bob

$Y_{j+1}, y_{j+1}, X_{i+1}$

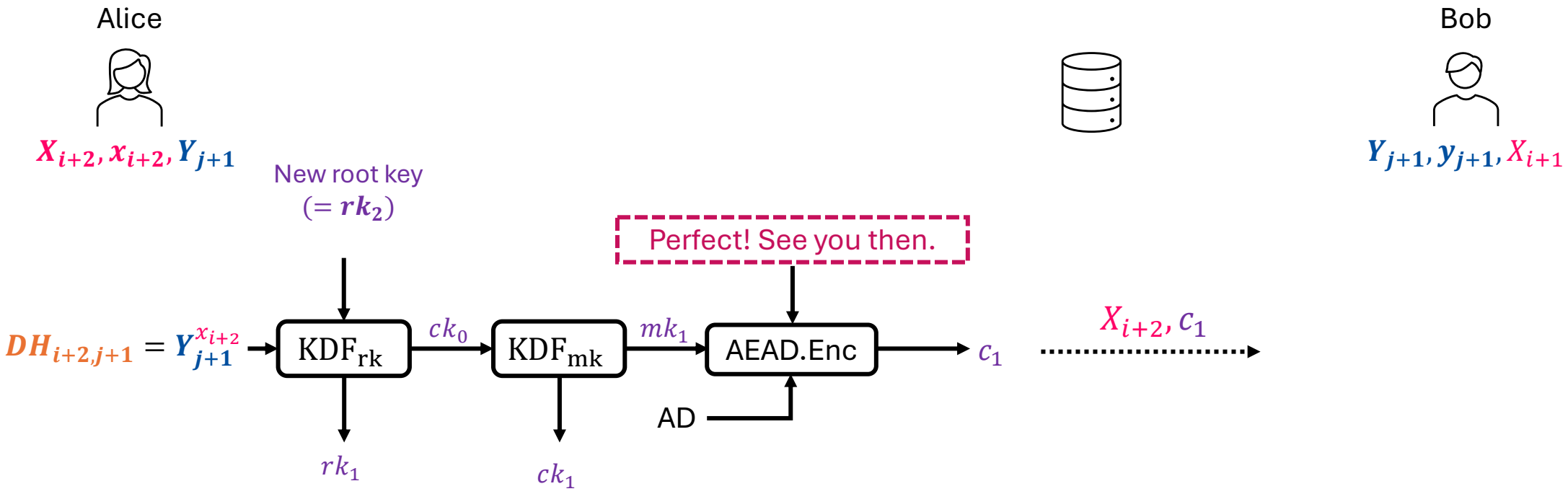
$DH_{i+2,j+1} = Y_{j+1}^{x_{i+2}}$

After decrypting Bob's messages, Alice

(1) Use the rk_2 as the new root key

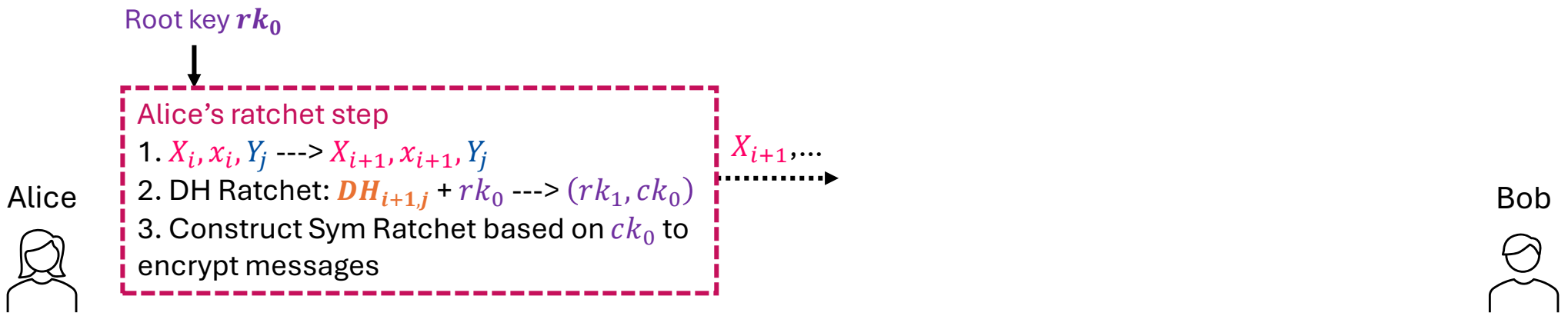
(2) Sample new ephemeral key pair

Double Ratchet



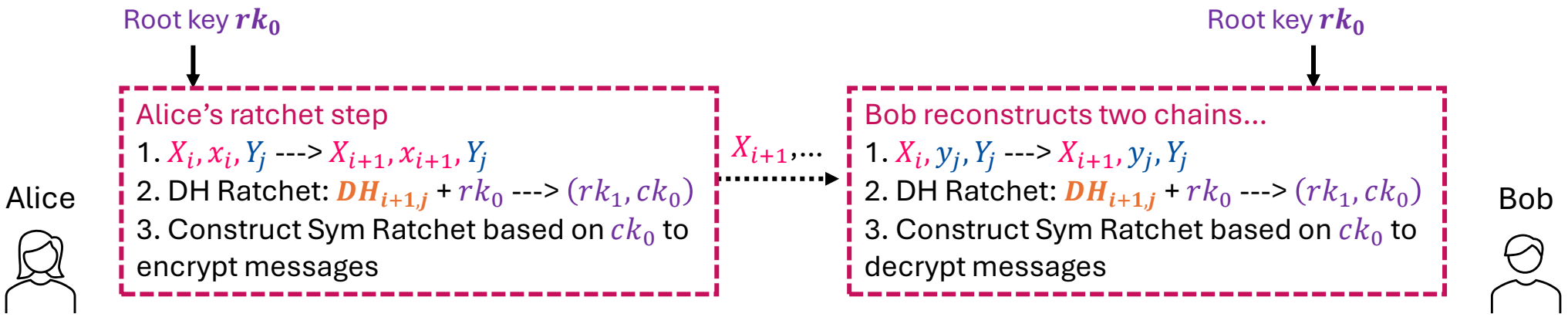
Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet



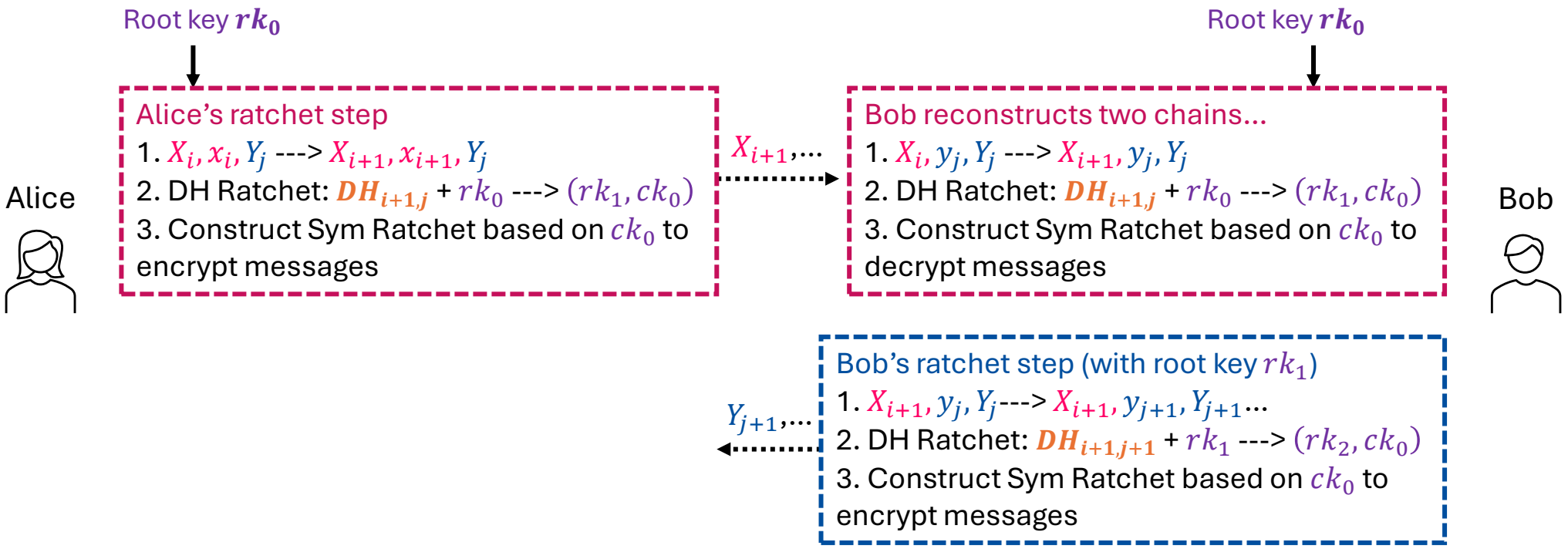
Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet



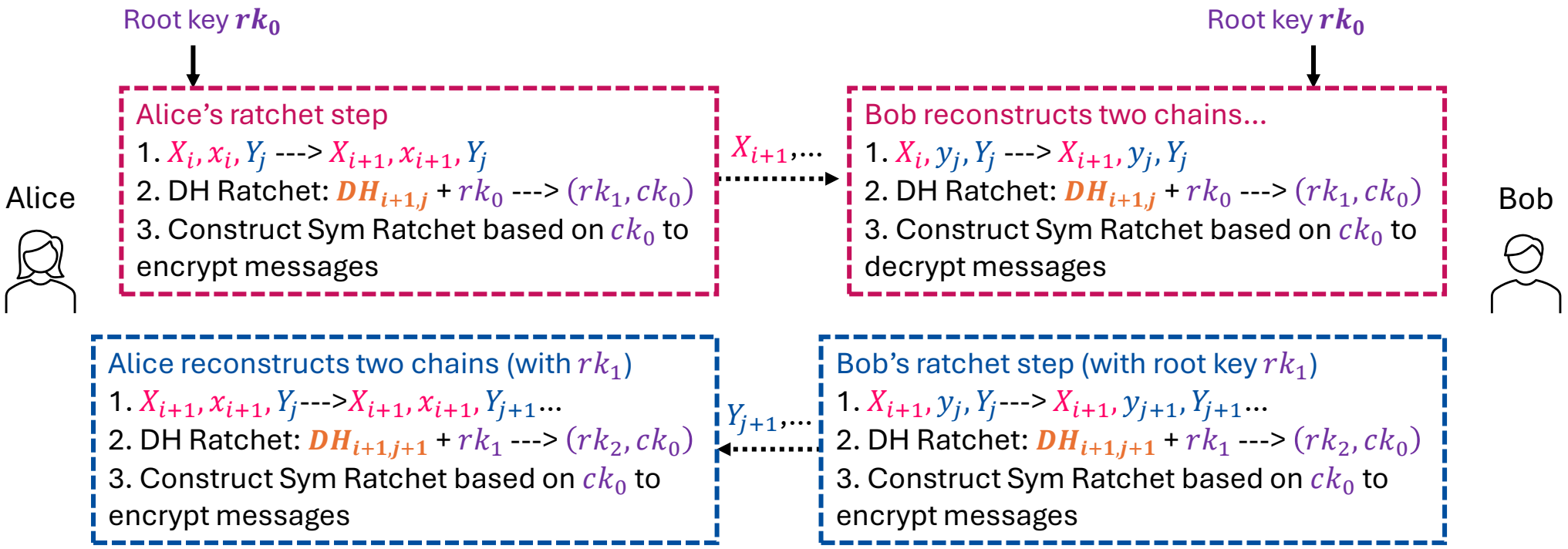
Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet



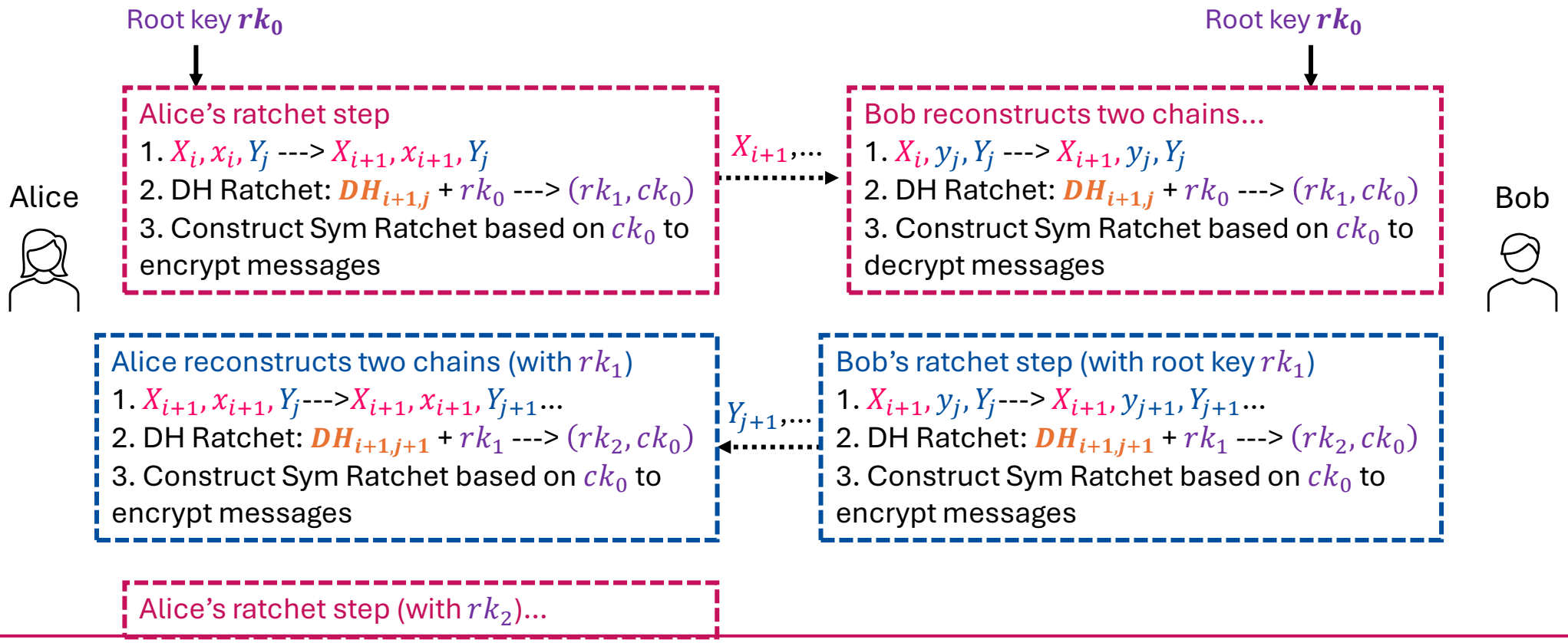
Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet



Double Ratchet

- The main idea: Symmetric-key Ratchet + Diffie-Hellman Ratchet



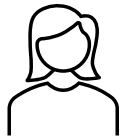
X3DH + Double Ratchet

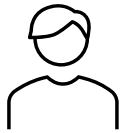
- Integrate Double Ratchet algorithm with X3DH
  - Use X3DH to bootstrap Double Ratchet
  - The Double Ratchet plays the role of a ‘post-X3DH’ protocol…

X3DH + Double Ratchet

- Recall of X3DH:

Public parameters: $(\mathbb{G}, g, q)$:
A q-order EC group $\mathbb{G}$ with a generator g

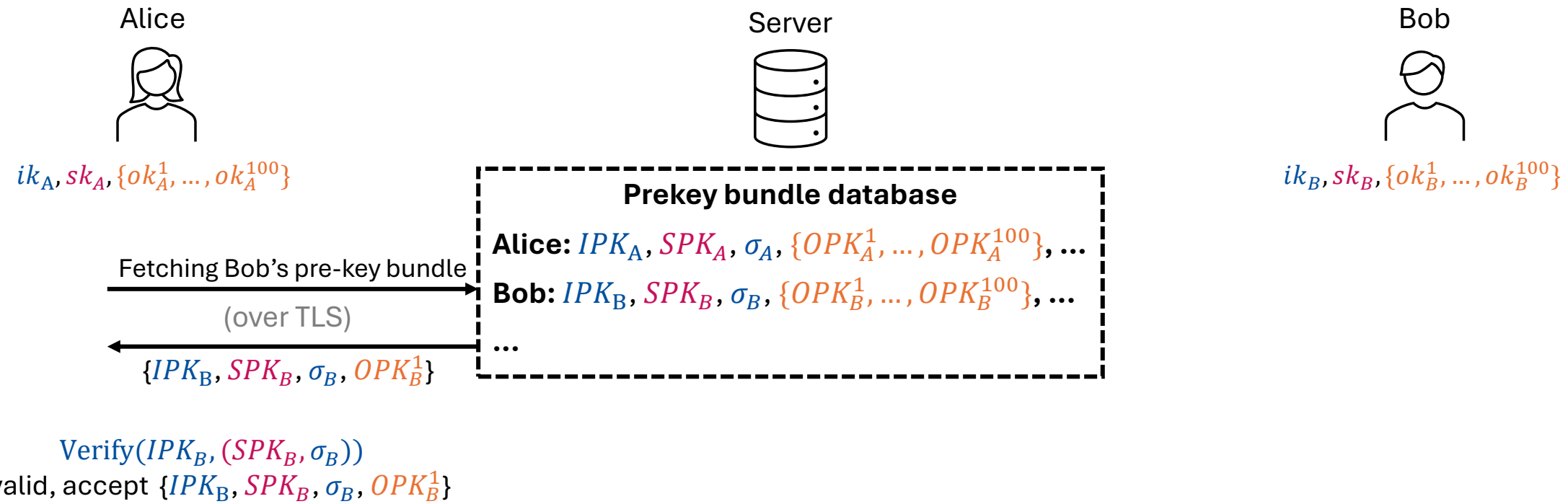

Alice


Bob

		Alice	Bob
Long-term secret (static)	Identity secret key (IK)	$ik_{\mathrm{A}} \in_{\$} \mathbb{Z}_q$	$ik_{\mathrm{B}} \in_{\$} \mathbb{Z}_q$
	Identity public key (IPK)	$IPK_{\mathrm{A}} (= g^{ik_A})$	IPK_{B}
Mid-term secret (updated periodically)	**Signing secret pre-key (SK)**	$\boldsymbol{sk_{\mathrm{A}}} \in_{\$} \mathbb{Z}_{\boldsymbol{q}}$	$\boldsymbol{sk_{\mathrm{B}}} \in_{\$} \mathbb{Z}_{\boldsymbol{q}}$
	Signing public pre-key (SPK)	$\boldsymbol{SPK_{\mathrm{A}}}$	$\boldsymbol{SPK_{\mathrm{B}}}$
Short-term secret (used once)	One-time secret pre-keys (OK)	$\{ok_{\mathrm{A}}^1, ok_A^2, \ldots\} \subseteq_{\$} \mathbb{Z}_q$	$\{ok_{\mathrm{B}}^1, ok_B^2, \ldots\} \subseteq_{\$} \mathbb{Z}_q$
	One-time public pre-keys (OPK)	$(OPK_{\mathrm{A}}^1, OPK_{\mathrm{A}}^2, \ldots)$	$(OPK_{\mathrm{B}}^1, OPK_{\mathrm{B}}^2, \ldots)$

X3DH + Double Ratchet

- Recall of X3DH:



$\text{Verify}(IPK_B, (SPK_B, \sigma_B))$
if valid, accept $\{IPK_B, SPK_B, \sigma_B, OPK_B^1\}$

X3DH + Double Ratchet

- X3DH:

Alice

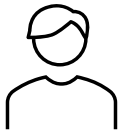
$ik_A, sk_A, \{ok_A^1, \dots, ok_A^{100}\}$

$\{IPK_B, SPK_B, OPK_B^1\}$

$ek_A \leftarrow_\$ \mathbb{Z}_q$

Server

Bob



$ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B^1)$

$IPK_A = g^{ik_A}, EPK_A = g^{ek_A}, \text{“1st OPK”}, \text{AEAD}_{KDF(SK_A)}(\text{msg} = \text{metadata}, \text{AD} = IPK_A || IPK_B)$

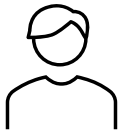
(Relayed by the server)

X3DH + Double Ratchet

- Initialize Double Ratchet using the SK from X3DH

Alice

Bob



$ik_A, sk_A, \{ok_A^1, \dots, ok_A^{100}\}$

$ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

$\{IPK_B, SPK_B, OPK_B^1\}$

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B^1)$

$IPK_A = g^{ik_A}, EPK_A = g^{ek_A}, \text{“1st OPK”}, \text{AEAD}_{KDF(SK_A)}(\text{msg} = \text{metadata}, \text{AD} = IPK_A || IPK_B)$

Initial root key $rk_0 = SK_A$

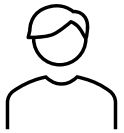
$X_0 = \perp, x_0 = \perp, Y_0 = SPK_B$

X3DH + Double Ratchet

- Initialize Double Ratchet using the SK from X3DH

Alice

Bob



$ik_A, sk_A, \{ok_A^1, \dots, ok_A^{100}\}$

$ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

$\{IPK_B, SPK_B, OPK_B^1\}$

$SK_A = \text{X3DH\_Key\_Alice}(ik_A, ek_A, IPK_B, SPK_B, OPK_B^1)$

$IPK_A = g^{ik_A}, EPK_A = g^{ek_A}, \text{“1st OPK”}, \text{AEAD}_{KDF(SK_A)}(\text{msg} = \text{metadata}, \text{AD} = IPK_A || IPK_B)$

Initial root key $rk_0 = SK_A$

$X_0 = \perp, x_0 = \perp, Y_0 = SPK_B$

$X_1 = g^{x_1}, x_1 \leftarrow_\$ \mathbb{Z}_q$

$(rk_1, ck_0) = \text{KDF}_{ck}(rk_0, \text{DH} = Y_0^{x_1})$

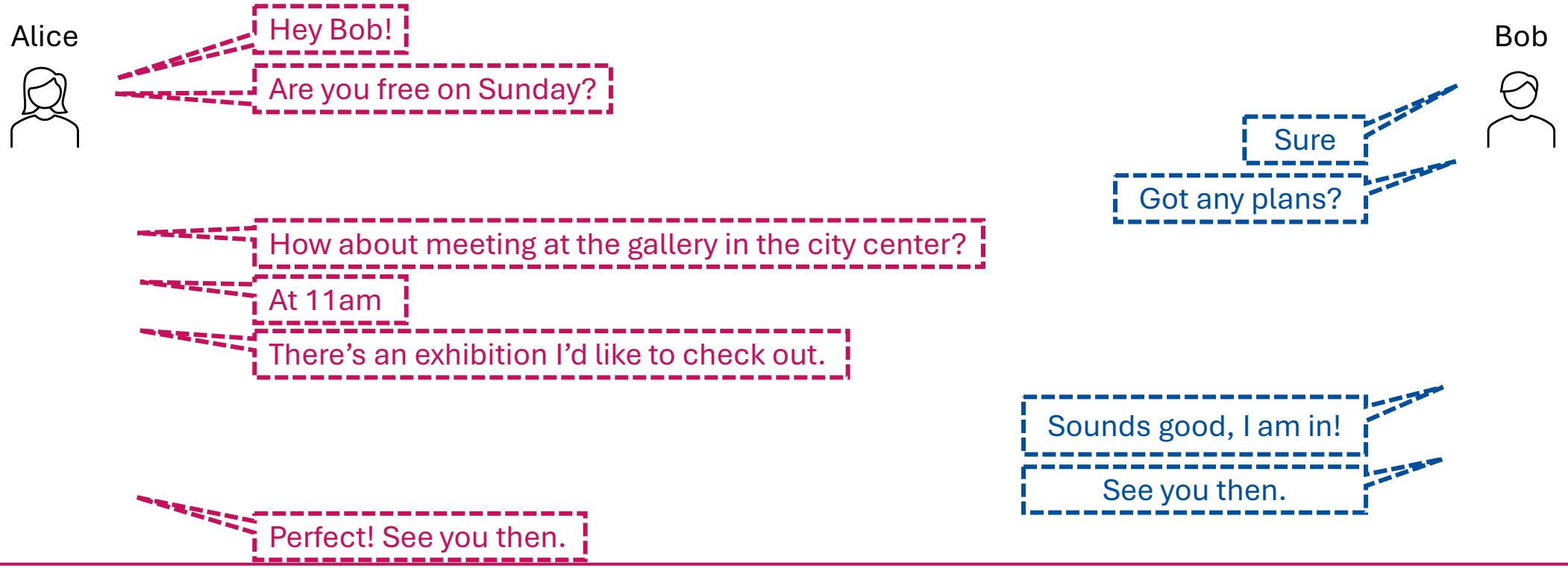
Then Alice uses ck_0 to construct the symmetric chain to encrypt messages…

Signal Secure Messaging Protocol

- Some technical details we do not cover:
  - XEdDSA and VXEdDSA:
    - ➢ DH key pairs for key exchange and signature…
  - Header encryption:
    - ➢ Cannot tell which messages belong to which sessions, or the ordering of messages within a session…
  - Out-of-order messages:
  - Session management and asynchronous settings

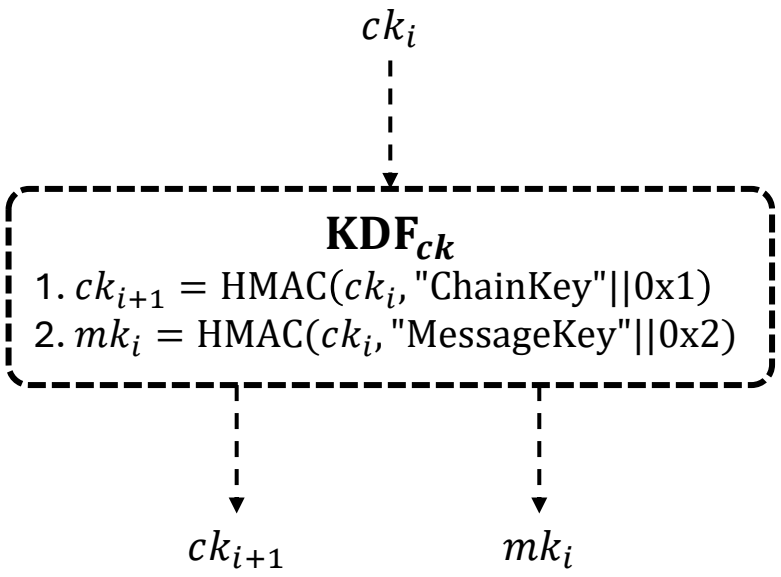
Exercise

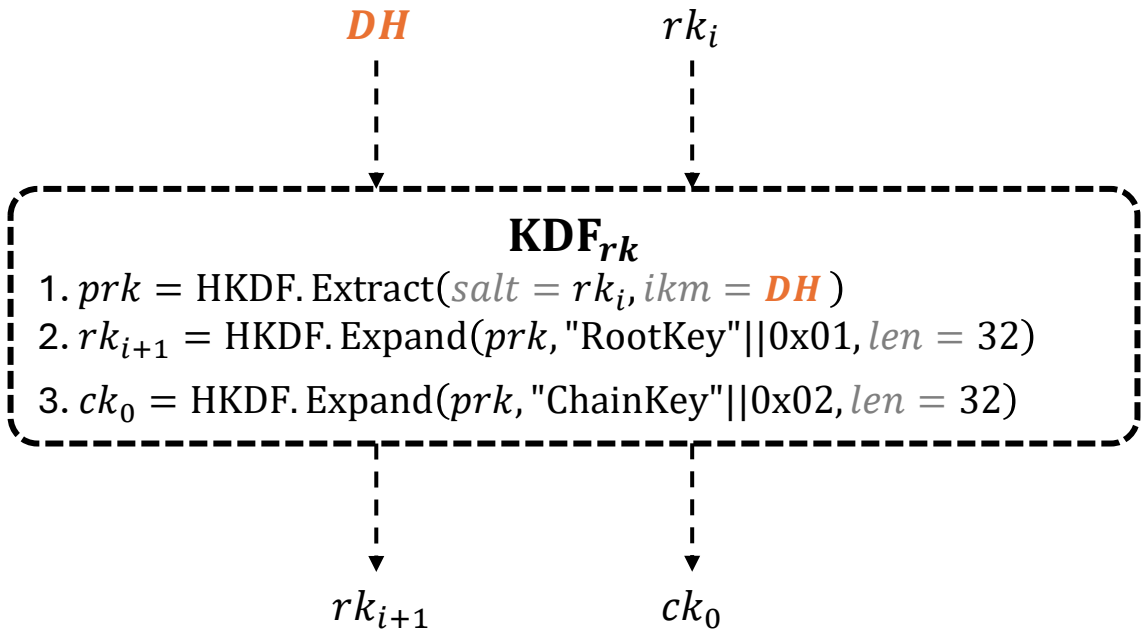
- (Without sockets) Use X3DH and Double Ratchet to encrypt this conversation (or you can choose other conversations):



Exercise

- The two chains in the last slide can implemented in the following way:





Further Reading

- Technical Documentations of Signal: https://signal.org/docs/
- Some research papers of analyzing security of Ratchet algorithms:
  - Bellare et al's work on formalizing ratcheted encryption/key exchange: https://eprint.iacr.org/2016/1028
  - Alwen et al's work on formalizing Double Ratchet: https://eprint.iacr.org/2018/1037
  - Collins et al's work on Tight security of Double Ratchet: https://eprint.iacr.org/2024/1625
  - …