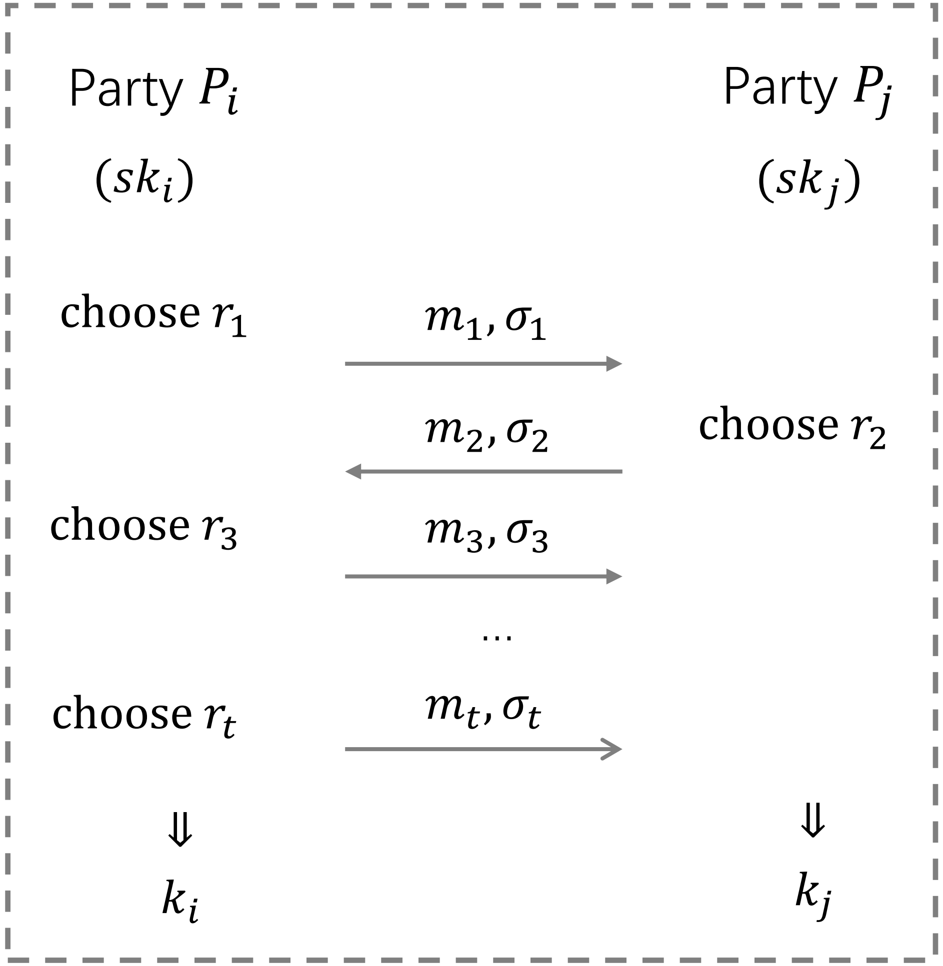
Two-Pass Authenticated Key Exchange with Explicit Authentication and Tight Security

Xiangyu Liu[1], Shengli Liu[1], Dawu Gu[1], and Jian Weng[2]

[1]**Shanghai Jiao Tong University, China**
[2]Jinan University, China

Authenticated Key Exchange (AKE)



A pass: one message sent from P_i to P_j (or P_j to P_i).

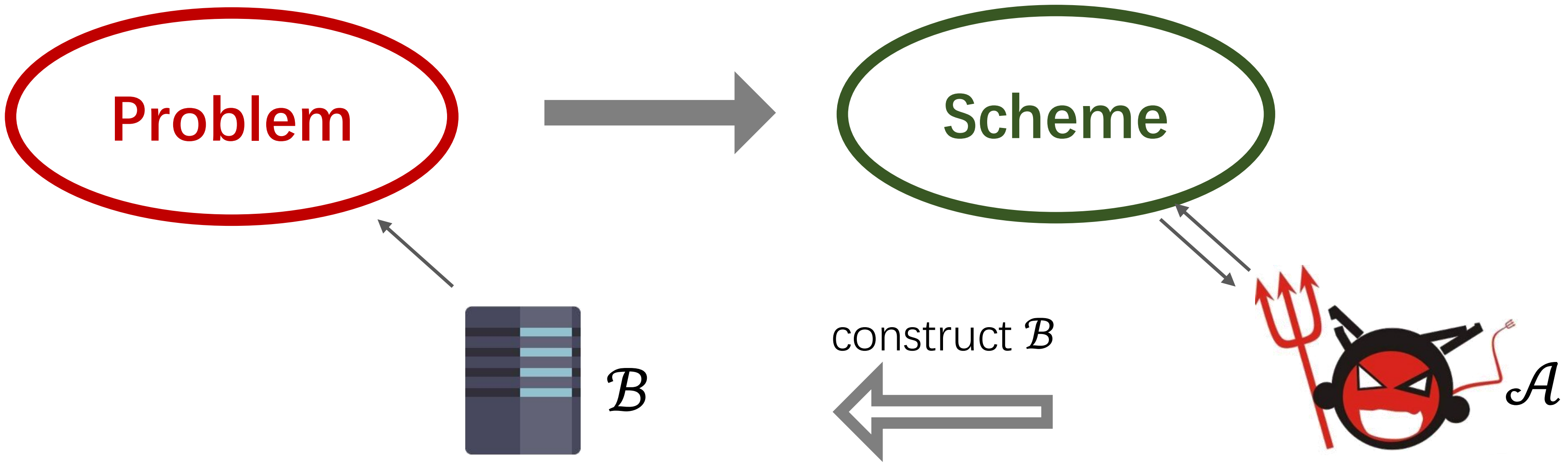
- **Correctness.**

  $$k_i = k_j.$$

- **Security.**
  - **Indistinguishability.**
    the session key is pseudorandom.
  - **Authentication.**
    - **Explicit authentication**: detects active attacks during the execution of AKE.
    - **Implicit authentication**: detects active attacks in the later communication.

Tight Security

Security of a cryptographic **Scheme** based on a hard **Problem**.



PPT algorithm $\mathcal{B}$ successfully solves **Problem** (with probability ϵ')

PPT adversary $\mathcal{A}$ successfully attacks **Scheme** (with probability ϵ)

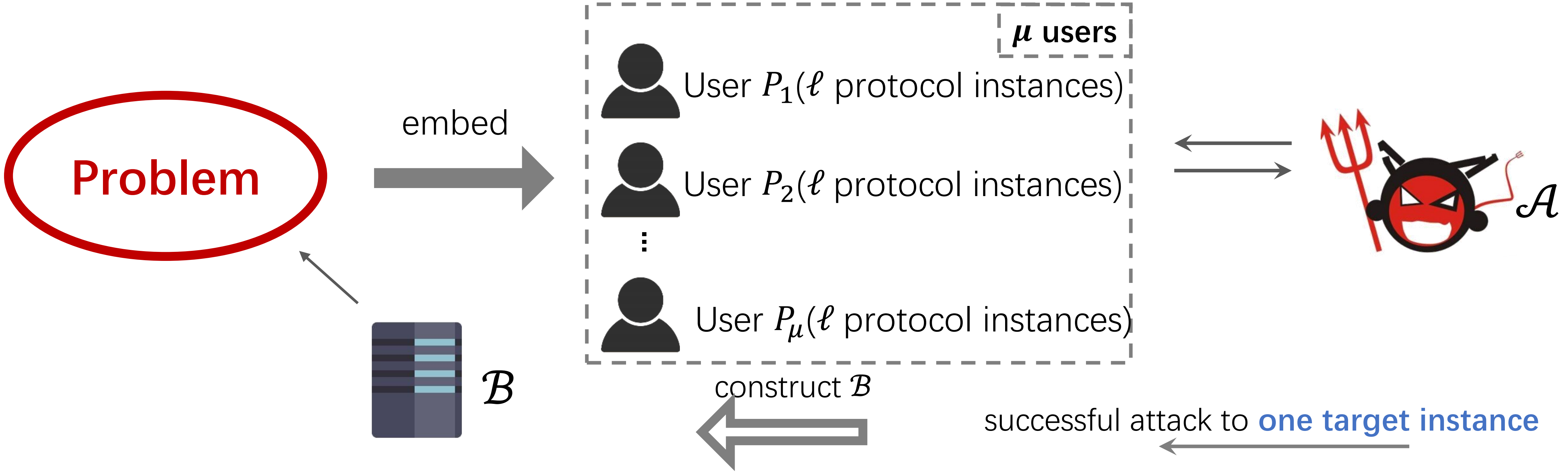
Security loss factor: $L = \frac{\epsilon}{\epsilon'}$

Tight Security: constant $L = O(1)$

Advantages:

- smaller elements
- universal key-length recommendations

Tight Security for AKE



PPT algorithm $\mathcal{B}$ successfully solves **Problem** (with probability ϵ')

PPT adversary $\mathcal{A}$ successfully attacks **AKE** (with probability ϵ)

Loose Security: loss factor at least $L = O(\mu\ell)$
Tight Security: constant $L = O(1)$

$\mu\ell$ can be as large as 2^{30}~2^{50}!

Related Works on Tightly Secure AKE

➢ Explicit authentication

- [GJ18, CRYPTO]: 3-pass protocol in the RO model.
- [BHJ+15, TCC]: 3-pass protocol in the Std. model.

Advantages of explicit authentication:

detect active attacks immediately.

➢ Implicit authentication

- [CCG+19, CRYPTO]: 2-pass protocol in the RO model (security loss $L = O(\mu)$).
- [XZM20, CT-RSA]: 2-pass protocol in the RO model.

2-pass AKE scheme with **explicit authentication** and **tight security**?

Security Model for AKE [GJ18]

$\mathcal{C}$

$(\pi_1^1, \dots, \pi_i^s, \dots, \pi_\mu^\ell)$

$\pi_i^s \rightleftarrows \pi_j^t$ …

$\mathcal{C}$ simulates their communication via $\mathcal{A}$'s **send** queries

$\mathcal{A}$

Send(i, s, msg) ←

response msg' →

Corrupt(i) ←

P_i's long-term sk_i →

RegisterCorrupt(i, pk_i) ←

Reveal(i, s) ←

session key k_i^s →

test(i, s) ←

independent random bit b_i^s

k_i^s/random key →

i^*, s^*, b' ←

($\mathcal{A}$'s guess of $b_{i^*}^{s^*}$ for target session (i^*, s^*))

Indistinguishability:

$$\Pr\left[b' = b_{i^*}^{s^*}\right] = \frac{1}{2} + \mathrm{negl}.$$

- μ: max number of users.
- ℓ: max number of executions per user involved.
- π_i^s: the (simulated) s-th of user P_i.

Signed Diffie-Hellman Protocol

Party P_i (sk_i)		Party P_j (sk_j)
$\sigma_1 \leftarrow \mathrm{SIG.Sign}(g^a)$	g^a, σ_1 $\longrightarrow$	
	$\longleftarrow$ g^b, σ_2	$\sigma_2 \leftarrow \mathrm{SIG.Sign}(g^a \| \| g^b)$
$\Downarrow$		$\Downarrow$
$k_i = g^{ab}$		$k_j = g^{ab}$

Commitment Problem in Signed DH

Hardness of tight security for signed DH.

Consider the reduction algorithm $\mathcal{B}$ and a specific session (i, s).

- $\mathcal{B}$ receives a DDH challenge problem (g^x, g^y, g^z).
- If (g^x, g^y, g^z) is embedded into session (i, s), then **it cannot be revealed**.
- If not, then $\mathcal{B}$ **cannot complete the reduction** if $\mathcal{A}$ chooses (i, s) as target.

Guess the target session (from $\mu\ell$ sessions) and embed the DDH problem into it.

$\Longrightarrow$ **loose security loss** $L = O(\mu\ell)$.

- To deal with the "commitment problem", Gjøsteen and Jager [CRYPTO 2018] added an extra hash commitment as the first message, resulting in a **3-pass** protocol with tight security in the RO model.

Commitment Problem in KEM

Key Encapsulation Mechanism (KEM):

- $\mathrm{KEM.Gen}: pk = g^a, sk = a$
- $\mathrm{KEM.Encap}(pk): K = g^{ab}, C = g^b$
- $\mathrm{KEM.Decap}(sk, C):\ K' = C^{sk}$

Signed DH protocol is actually a **KEM + SIG** construction.

We need to solve the **commitment problem in KEM**:

- provide traditional IND-security
- answer reveal queries from $\mathcal{A}$

Our Solution: IND-mCPA$^{\text{reveal}}$ secure KEM

IND-mCPA$^{\text{reveal}}$ security experiment:

$\mathcal{C}$		$\mathcal{A}$
For $i \in [\mu]$: $(pk_i, sk_i) \leftarrow \text{KEM.Gen}$	$\{pk_i\}$ →	
$(K_0, C) \leftarrow \text{KEM.Encap}, K_1 \leftarrow \$$ $\beta \leftarrow \{0,1\}$ add (i, C, β) to CList	← **Encap**(i) K_β, C →	(challenge ciphertexts)
$K' \leftarrow \text{KEM.Decap}(sk_i, C')$ add (i, C') to RList	← **Reveal**(i, C') K' →	
$\mathcal{A}$ *wins* if $\exists (i^*, C^*, \beta) \in \text{CList}$ $s.t. (i^*, C^*) \notin \text{RList} \wedge \beta' = \beta$	← i^*, C^*, β'	

IND-mCPA$^{\text{reveal}}$ security: $\Pr[\mathcal{A}\ wins] = \frac{1}{2} + \text{negl.}$

Our Solution: MU-EUF-CMA$^{\text{corr}}$ secure SIG

MU-EUF-CMA$^{\text{corr}}$ security experiment:

$\mathcal{C}$ | $\mathcal{A}$

$For\ i \in [\mu]$:
$(vk_i, sk_i) \leftarrow \mathrm{SIG.Gen}$

$\{vk_i\}$ →

← **Sign**(i, m)

$\sigma \leftarrow \mathrm{SIG.Sign}(sk_i, m)$

add (m, σ) to S_i

σ →

← **Corrupt**(i)

add (i, C') to S^{corr}

sk_i →

← i^*, m^*, σ^*

$\mathcal{A}\ wins$ if $i^* \notin S^{\mathrm{corr}} \wedge (m^*, \cdot) \notin S_{i^*}$
$\wedge\ \mathrm{SIG.Ver}(vk_{i^*}, m^*, \sigma^*) = 1$

MU-EUF-CMA$^{\text{corr}}$ security: $\Pr[\mathcal{A}\ wins] = \mathrm{negl.}$

Our Construction: KEM + SIG

- With a tightly IND-mCPA$^{\text{reveal}}$ secure KEM, the commitment problem is solved, since all challenge ciphertexts can be
  - either served as the final target of $\mathcal{A}$.
  - or revealed to $\mathcal{A}$.
- With a tightly MU-EUF-CMA$^{\text{corr}}$ secure SIG, we can also handle the corruption queries from the adversary.

✓ KEM: tightly IND-mCPA$^{\text{reveal}}$ security → indistinguishability

✓ SIG: tightly MU-EUF-CMA$^{\text{corr}}$ security → explicit authentication

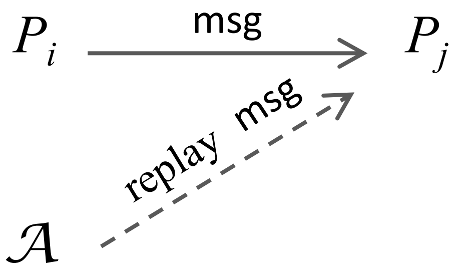
Our Construction: KEM + SIG

Party P_i		Party P_j
(sk_i)		(sk_j)
$(pk_{\mathrm{KEM}}, sk_{\mathrm{KEM}}) \leftarrow \mathrm{KEM.Gen}$ $\sigma_1 \leftarrow \mathrm{SIG.Sign}(sk_i, pk_{\mathrm{KEM}})$	$(pk_{\mathrm{KEM}}, \sigma_1)$ $\longrightarrow$	
		$(K, C) \leftarrow \mathrm{KEM.Encap}(pk_{\mathrm{KEM}})$ $\sigma_2 \leftarrow \mathrm{SIG.Sign}(sk_j, pk_{\mathrm{KEM}} \| C)$
$K' \leftarrow \mathrm{KEM.Decap}(sk_{\mathrm{KEM}}, C)$	$\longleftarrow$ (C, σ_2)	
$\Downarrow$ $k_i = K'$		$\Downarrow$ $k_j = K$

Against $\mathcal{A}$'s queries (attacks):

- **Corrupt**: SIG is secure against adaptive corruptions.
- **Reveal**: KEM is secure against adaptive reveals.
- **Test**: KEM is IND-secure.

Dealing with Replay Attacks

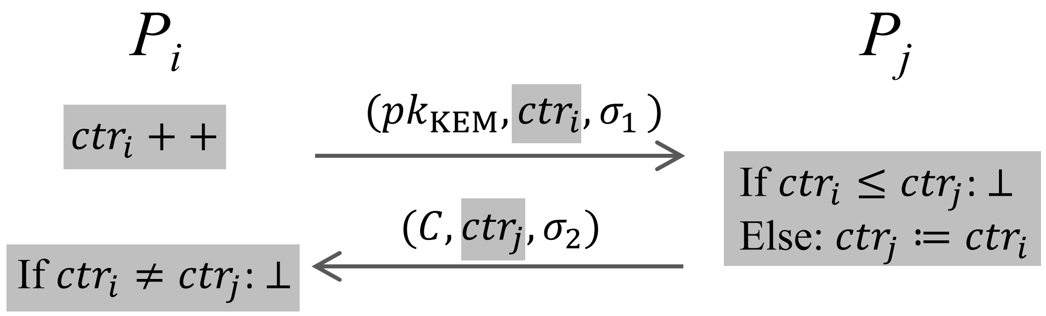


Compared with multi-pass AKE, 2-pass AKE inherently open to replay attacks.

- **A stronger security model of AKE**:

  If a replayed message is accepted by some user, the authentication of AKE is broken.

- **We add counters to identify the freshness of messages.**



✓ In this way, any replayed attacks can be detected immediately in our 2-pass AKE.

Our Generic Construction

Party P_i		Party P_j
(sk_i)		(sk_j)
$(pk_{\mathrm{KEM}}, sk_{\mathrm{KEM}}) \leftarrow \mathrm{KEM.Gen}$ $ctr_i ++$ $\sigma_1 \leftarrow \mathrm{SIG.Sign}(sk_i, pk_{\mathrm{KEM}} \| ctr_i)$	$(pk_{\mathrm{KEM}}, ctr_i, \sigma_1)$ $\longrightarrow$	
		check $ctr_i > ctr_j$? If yes, $ctr_j := ctr_i$ $(K, C) \leftarrow \mathrm{KEM.Encap}(pk_{\mathrm{KEM}})$ $\sigma_2 \leftarrow \mathrm{SIG.Sign}(sk_j, pk_{\mathrm{KEM}} \| ctr_j \| C)$
check $ctr_i = ctr_j$? $K' \leftarrow \mathrm{KEM.Decap}(sk_{\mathrm{KEM}}, C)$	$\longleftarrow$ (C, ctr_j, σ_2)	
$\Downarrow$ $k_i = K'$		$\Downarrow$ $k_j = K$

- ✓ **Perfect Forward Security**
- ✓ **KCI Resistance** (security against key-compromise impersonation attacks)

AKE in the RO model

➢ Instantiation of KEM

$\mathrm{KEM}_{\mathrm{st2DH}}$:

- $\mathrm{KEM.Gen}: pk = (g^{x_1}, g^{x_2}), sk = (x_1, x_2)$.
- $\mathrm{KEM.Encap}(pk): K = H(pk, C, g^{x_1 y}, g^{x_2 y}), C = g^y$
- $\mathrm{KEM.Decap}\big((x_1, x_2), C\big): K' = H(pk, C, C^{x_1}, C^{x_2})$

- The IND-mCPA$^{\text{reveal}}$ security is based on the twin DH assumption (the CDH assumption).
- Tight security relies on the random self-reducibility.
- Security against reveal queries relies on the decisional oracle $2DH$.

➢ Instantiation of SIG

- $\mathrm{SIG}_{\mathrm{DDH}}$ in [GJ18] (based on the DDH assumption).

We obtain the first **2-pass** AKE scheme with **explicit authentication** and **tight security** in the RO model.

AKE in the Std. model

➢ Instantiation of KEM

- $\mathrm{KEM}_{\mathrm{MDDH}}$ is derived from the tightly IND-mCCA secure PKE scheme by Han et al. [CRYPTO 2019].
- **IND-mCCA implies IND-mCPA$^{\text{reveal}}$ with tight reduction.**

➢ Instantiation of SIG

- $\mathrm{SIG}_{\mathrm{MDDH}}$ in [BHJ+15] (based on the MDDH assumption).

We obtain the first **2-pass** AKE scheme with **explicit authentication** and **tight security** in the Std. model.

Comparison

AKE Scheme	Comp. (I)	Comp. (R)	Comm. (I+R)	Assumption	Sec. Loss	#Pass	Model
[GJ18]	17	17	12+11	DDH	$O(1)$	**3**	RO
AKE$_{\text{DDH}}$	19	18	12+11	DDH	$O(1)$	**2**	RO
[BHJ+15]	22 $O(k^2)$	23 $O(k^2)$	11+9 $(2k^2 + 4k + 5) + (4k + 7)$	1-LIN=SXDH D_k-MDDH	$O(\lambda)$	**3**	Std.
AKE$_{\text{MDDH}}$	37 $O(k^3)$	22 $O(k^3)$	7+8 $(k^2 + 5k + 1) + (4k + 4)$	1-LIN=SXDH D_k-MDDH	$O(\lambda)$	**2**	Std.

Conclusion

stronger security model
(covers replay attacks)

⇩

tightly secure
AKE

tightly IND-mCPA$^{\text{reveal}}$
KEM

tightly EUF-CMA$^{\text{corr}}$
SIG

instantiations

RO: $\mathrm{KEM}_{\mathrm{st2DH}}$

Std.: $\mathrm{KEM}_{\mathrm{MDDH}}$

RO: $\mathrm{SIG}_{\mathrm{DDH}}$

Std.: $\mathrm{SIG}_{\mathrm{MDDH}}$

$\mathrm{AKE}_{\mathrm{DDH}}$ (RO model)

$\mathrm{AKE}_{\mathrm{MDDH}}$ (Std. model)

- 2-pass
- explicit authentication
- tight security

Thank you!
Questions?