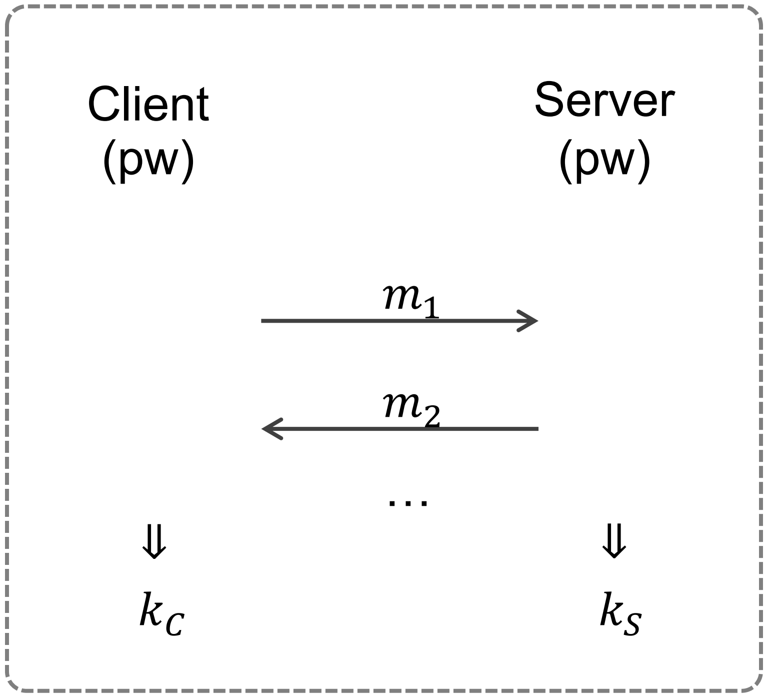
EKE Meets Tight Security in the Universally Composable Framework

Xiangyu Liu, Shengli Liu, Shuai Han, Dawu Gu
Shanghai Jiao Tong University

May 9, 2023

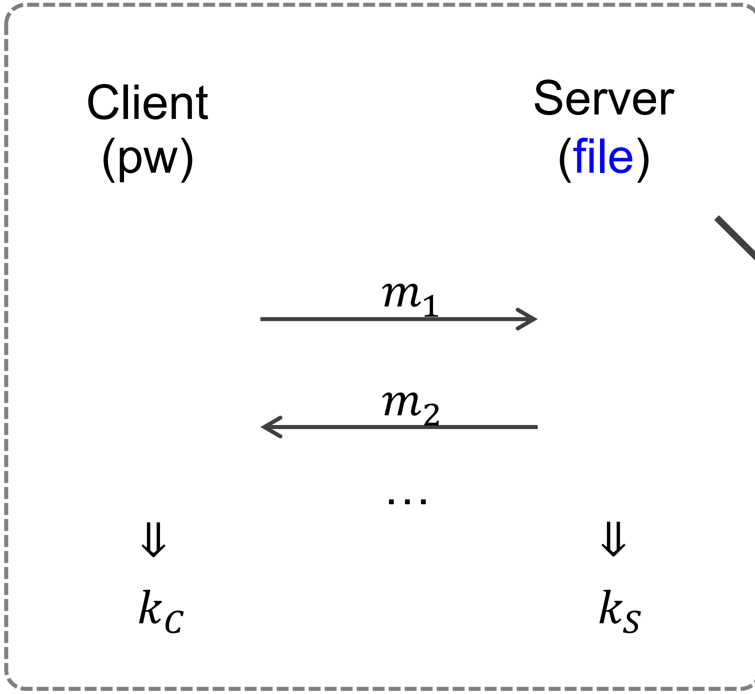
PAKE

➢ PAKE: password-based authenticated key exchange



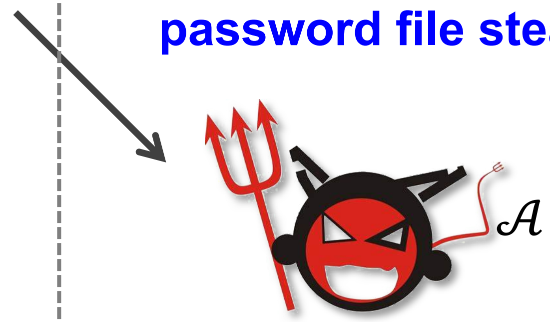
- Password (pw) is short and human-memorable
- No (complicated) cryptographic keys

Asymmetric PAKE (aPAKE)



potential attacks:

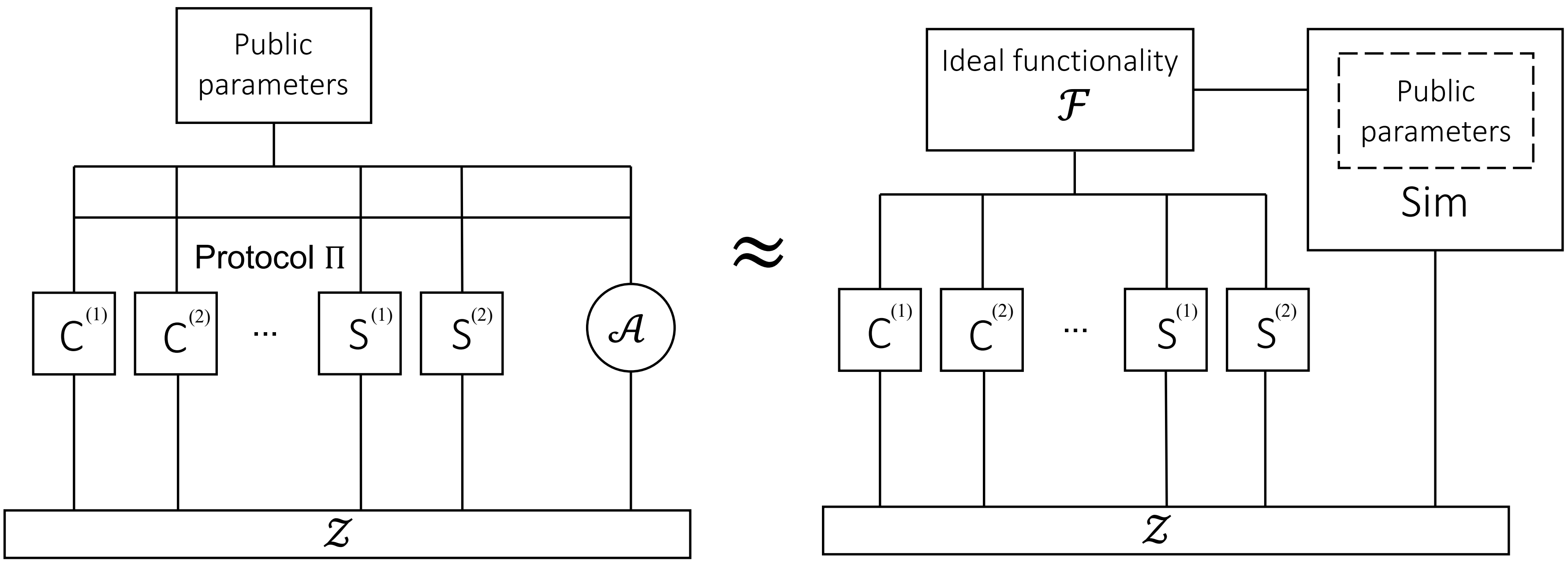
password file stealing



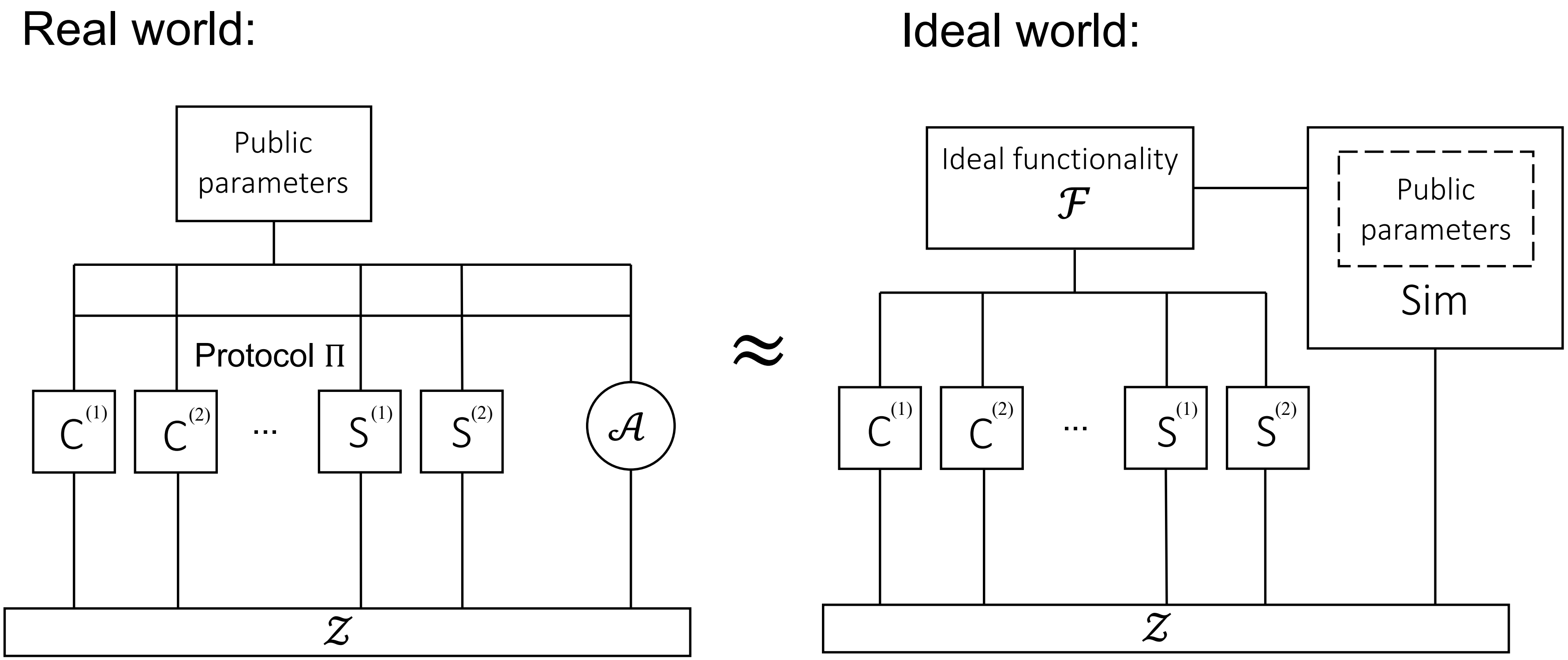
- Password file: a hash value of pw (e.g., $H(\text{pw})$)
- Prevent Adversary (with file) from impersonating Client to log in Server

Universally Composable (UC) framework

Real world: Ideal world:



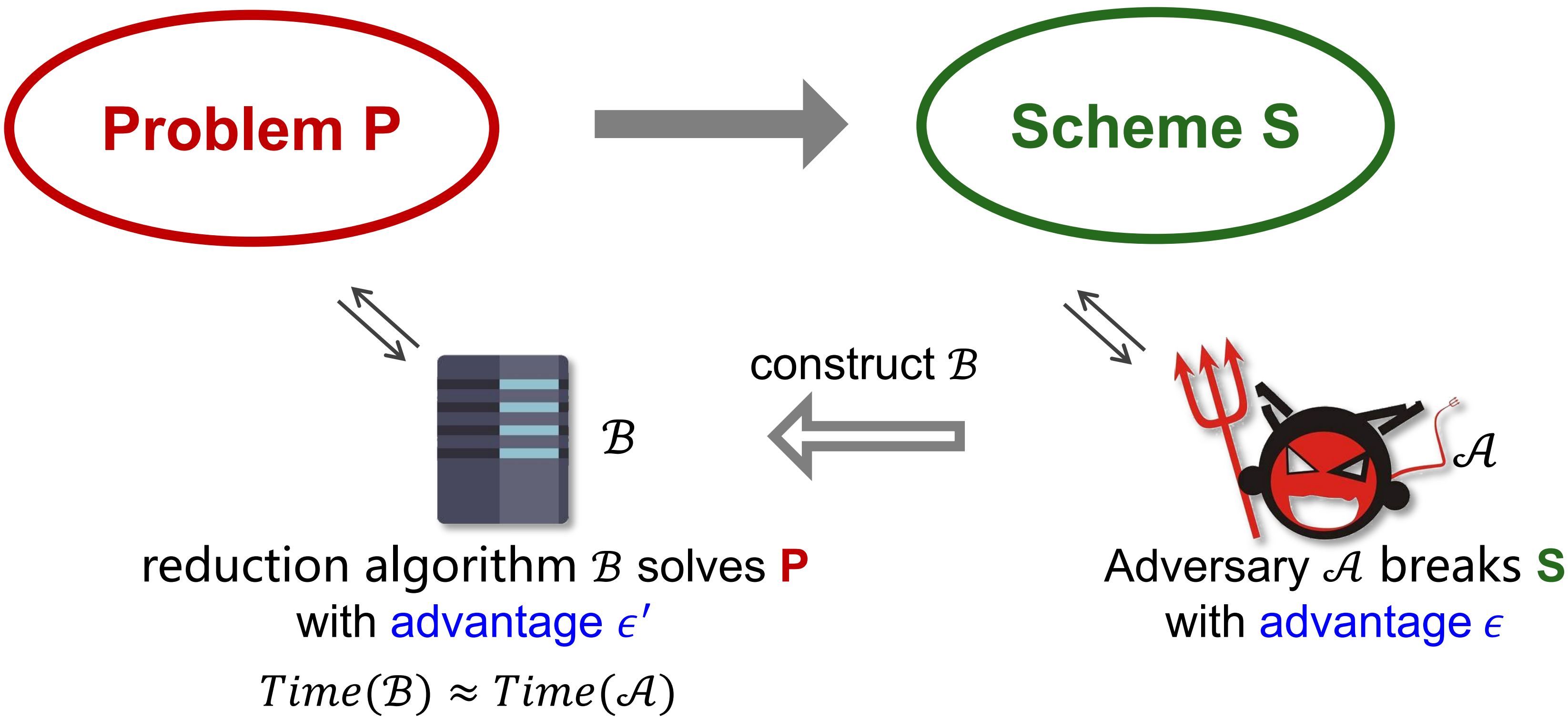
Universally Composable (UC) framework



Advantages of UC security model:

- Arbitrary correlation and distributions for pw
- Universal composition theorem is appliable
  (security preserves even running in arbitrary networks)

Provable security: reduction



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

Tight security: $L = O(1)$ or $L = \text{Poly}(\lambda)$

Loose security: L depends on $\mathcal{A}$'s behaviors

Advantages of tight security

Hybrid argument:

security in the **single** user/session setting $\Longrightarrow$ security in the **multi-** user/session setting

------ Cost of huge security loss!! (as high as $2^{30} \sim 2^{50}$)

Advantages of tight security

Hybrid argument:

security in the **single** user/session setting $\Longrightarrow$ security in the **multi-** user/session setting

------ Cost of huge security loss!! (as high as $2^{30} \sim 2^{50}$)

Tight security

- ✓ universal parameters
- ✓ smaller parameters (under the same security level)

YES!!

Related works

Only 2 related works about tightly secure (a)PAKE

- [BIO+17]：IND model (**weaker than** UC model), Gap DH assumption, PAKE protocol
- [ABB+20]：relaxed UC model, Gap DH assumption, PAKE protocol

Gap DH assumption (**non-standard** interactive assumption):

------ Given (g, g^x, g^y) and a decisional oracle for DDH tuples, computing g^{xy} is hard

[BIO+17] Becerra, J., Iovino, V., Ostrev, D., Sala, P., Skrobot, M.: Tightly-secure PAK(E).

[ABB+20] Abdalla, M., Barbosa, M., Bradley, T., Jarecki, S., Katz, J., Xu, J.: Universally composable relaxed password authenticated key exchange.

Related works

Only 2 related works about tightly secure (a)PAKE

- [BIO+17] : IND model (**weaker than** UC model), Gap DH assumption, PAKE protocol
- [ABB+20] : relaxed UC model, Gap DH assumption, PAKE protocol

Gap DH assumption (**non-standard** interactive assumption):

------ Given (g, g^x, g^y) and a decisional oracle for DDH tuples,

computing g^{xy} is hard

(a)PAKE protocols with tight security in UC framework, from standard hardness assumptions?

[BIO+17] Becerra, J., Iovino, V., Ostrev, D., Sala, P., Skrobot, M.: Tightly-secure PAK(E).

[ABB+20] Abdalla, M., Barbosa, M., Bradley, T., Jarecki, S., Katz, J., Xu, J.: Universally composable relaxed password authenticated key exchange.

Contributions

- 2DH-EKE protocol (PAKE)

  ------ based on CDH assumption, tight UC security

- Negative result for the tight security of aPAKE

  ------ lower bound: N (total number of “Client-Server” pairs)

- 2DH-aEKE protocol (aPAKE)

  ------ based on CDH assumption, UC security, optimal security loss N

EKE (Encrypted Key Exchange) protocol

Client (pw)		Server (pw)
$x \leftarrow \mathbb{Z}_q$		$y \leftarrow \mathbb{Z}_q$
$e_1 \leftarrow \mathrm{E}_1(\mathrm{pw}, X = g^x)$	e_1 →	$e_2 \leftarrow \mathrm{E}_2(\mathrm{pw}, Y = g^y)$
	← e_2	
$k_C = H(trans, Z = g^{xy})$		$k_S = H(trans, Z = g^{xy})$

Reduction algorithm $\mathcal{B}$: randomize the CDH/DDH challenge problem, and embed them into multiple session instances (random self-reducibility of the DH problem)

[BM92] Bellovin, S.M., Merritt, M.: Encrypted key exchange: password-based protocols secure against dictionary attacks.

EKE (Encrypted Key Exchange) protocol

Client (pw)		Server (pw)
$x \leftarrow \mathbb{Z}_q$		$y \leftarrow \mathbb{Z}_q$
$e_1 \leftarrow \mathrm{E}_1(\mathrm{pw}, X = g^x)$	$\xrightarrow{e_1}$	$e_2 \leftarrow \mathrm{E}_2(\mathrm{pw}, Y = g^y)$
	$\xleftarrow{e_2}$	
$k_C = H(trans, Z = g^{xy})$		$k_S = H(trans, Z = g^{xy})$

Reduction algorithm $\mathcal{B}$: randomize the CDH/DDH challenge problem, and embed them into multiple session instances (random self-reducibility of the DH problem)

Obstacles:

1. If $\mathcal{A}$ attacks successfully (computes Z correctly and queries $H(trans, Z)$), how can $\mathcal{B}$ extract the correct CDH value?
2. If $\mathcal{A}$ guesses pw correctly (hence can compute key), how can $\mathcal{B}$ do?

[BM92] Bellovin, S.M., Merritt, M.: Encrypted key exchange: password-based protocols secure against dictionary attacks.

2DH decisional oracle

Strong Twin DH (st2DH) assumption：

- Given $(g^{x_1}, g^{x_2}, g^{y})$ and decisional oracle $\mathrm{2DH}(\cdot,\cdot,\cdot)$, computing $(g^{x_1 y}, g^{x_2 y})$ is hard
- $\mathrm{2DH}(\cdot,\cdot,\cdot)$ inputs $\left(g^{y'}, g^{z_1}, g^{z_2}\right)$, outputs whether $(x_1 y' = z_1) \wedge (x_2 y' = z_2)$

[CKS08]：st2DH assumption $\Longleftrightarrow$ CDH assumption

[CKS08] Cash, D., Kiltz, E., Shoup, V.: The twin diffie-hellman problem and applications.

Idea: 2DH-EKE protocol

Client (pw)		Server (pw)
$x_1, x_2 \leftarrow \mathbb{Z}_q$		$y \leftarrow \mathbb{Z}_q$
$e_1 \leftarrow \mathrm{E}_1(\mathrm{pw}, X_1 = g^{x_1}, X_2 = g^{x_2})$	$\xrightarrow{e_1}$	$e_2 \leftarrow \mathrm{E}_2(\mathrm{pw}, Y = g^{y})$
	$\xleftarrow{e_2}$	
$k_C = H(trans, Z_1 = g^{x_1 y}, Z_2 = g^{x_2 y})$		$k_S = H(trans, Z_1 = g^{x_1 y}, Z_2 = g^{x_2 y})$

Solve the two obstacles:

1. If $\mathcal{A}$ attacks successfully (computes Z correctly and queries $H(trans, Z)$), how can $\mathcal{B}$ extract the correct CDH value?

   ------ locate the correct Z_1, Z_2 via checking $2\mathrm{DH}(Y, Z_1, Z_2) == 1$?

2. If $\mathcal{A}$ guesses pw correctly (hence can compute key), how can $\mathcal{B}$ do?

   ------ $2\mathrm{DH}(\cdot,\cdot,\cdot)$ and the simulation of RO, keep $\mathcal{A}$'s view consistent.

Towards UC: ideal ciphers

Client (pw)		Server (pw)
$x_1, x_2 \leftarrow \mathbb{Z}_q$		$y \leftarrow \mathbb{Z}_q$
$e_1 \leftarrow \mathrm{E}_1(\mathrm{pw}, X_1 = g^{x_1}, X_2 = g^{x_2})$	$\xrightarrow{e_1}$	$e_2 \leftarrow \mathrm{E}_2(\mathrm{pw}, Y = g^{y})$
	$\xleftarrow{e_2}$	
$k_C = H(trans, Z_1 = g^{x_1 y}, Z_2 = g^{x_2 y})$		$k_S = H(trans, Z_1 = g^{x_1 y}, Z_2 = g^{x_2 y})$

Using ideal ciphers to achieve UC security:

- ✓ Simulate transcripts e_1, e_2 without pw
- ✓ Deduce the password guess (pw') in $\mathcal{A}$'s mind from IC list:
  - correct guess: honest execution and real session key
  - wrong guess: random key (security relies on st2DH assumption)

aPAKE: additional computation

2DH-aEKE protocol:

Client (pw)	$\text{file} = (\text{h}, V_1 = g^{v_1}, V_2 = g^{v_2})$	Server (file)
$(\text{h}, v_1, v_2) \leftarrow H_0(\text{pw})$ $x_1, x_2 \leftarrow \mathbb{Z}_q$	e_1 →	$y \leftarrow \mathbb{Z}_q$
$e_1 \leftarrow \text{E}_1(\text{h}, X_1 = g^{x_1}, X_2 = g^{x_2})$	← e_2	$e_2 \leftarrow \text{E}_2(\text{h}, Y = g^{y})$

$$Z_1 = g^{x_1 y}, Z_2 = g^{x_2 y}, Z_3 = g^{v_1 y}, Z_4 = g^{v_2 y}$$

$(k_C, \sigma) = H(trans, Z_1, Z_2, Z_3, Z_4, \text{h})$	σ →	$(k_S, \sigma') = H(trans, Z_1, Z_2, Z_3, Z_4, \text{h})$ $\sigma = \sigma'?$

MAC σ: achieve **perfect forward security**

aPAKE: additional computation

2DH-aEKE protocol:

Client (pw)	$\text{file} = (\text{h}, V_1 = g^{v_1}, V_2 = g^{v_2})$	Server (file)
$(h, v_1, v_2) \leftarrow H_0(\text{pw})$ $x_1, x_2 \leftarrow \mathbb{Z}_q$ $e_1 \leftarrow \text{E}_1(\text{h}, X_1 = g^{x_1}, X_2 = g^{x_2})$	e_1 →	$y \leftarrow \mathbb{Z}_q$
	← e_2	$e_2 \leftarrow \text{E}_2(\text{h}, Y = g^y)$

$$Z_1 = g^{x_1 y}, Z_2 = g^{x_2 y}, Z_3 = g^{v_1 y}, Z_4 = g^{v_2 y}$$

Client		Server
$(k_C, \sigma) = H(trans, Z_1, Z_2, Z_3, Z_4, \text{h})$	σ →	$(k_S, \sigma') = H(trans, Z_1, Z_2, Z_3, Z_4, \text{h})$ $\sigma = \sigma'$?

MAC σ: achieve **perfect forward security**

Optimal security loss for aPAKE

For aPAKE, **simple reductions** have an **optimal security loss N** (total number of "Client-Server" pairs).

Simple reduction: invoke Adversary only once

Conclusion

- 2DH-EKE protocol (PAKE)

  ------ based on CDH assumption, tight UC security
- Negative result for the tight security of aPAKE

  ------ lower bound: N (total number of “Client-Server” pairs)
- 2DH-aEKE protocol (aPAKE)

  ------ based on CDH assumption, UC security, optimal security loss N

Conclusion

- 2DH-EKE protocol (PAKE)
  ------ based on CDH assumption, tight UC security
- Negative result for the tight security of aPAKE
  ------ lower bound: N (total number of "Client-Server" pairs)
- 2DH-aEKE protocol (aPAKE)
  ------ based on CDH assumption, UC security, optimal security loss N

Thank you!

Xiangyu Liu (xiangyu1994liu@gmail.com)