SPHF-Friendly Non-Interactive Commitments

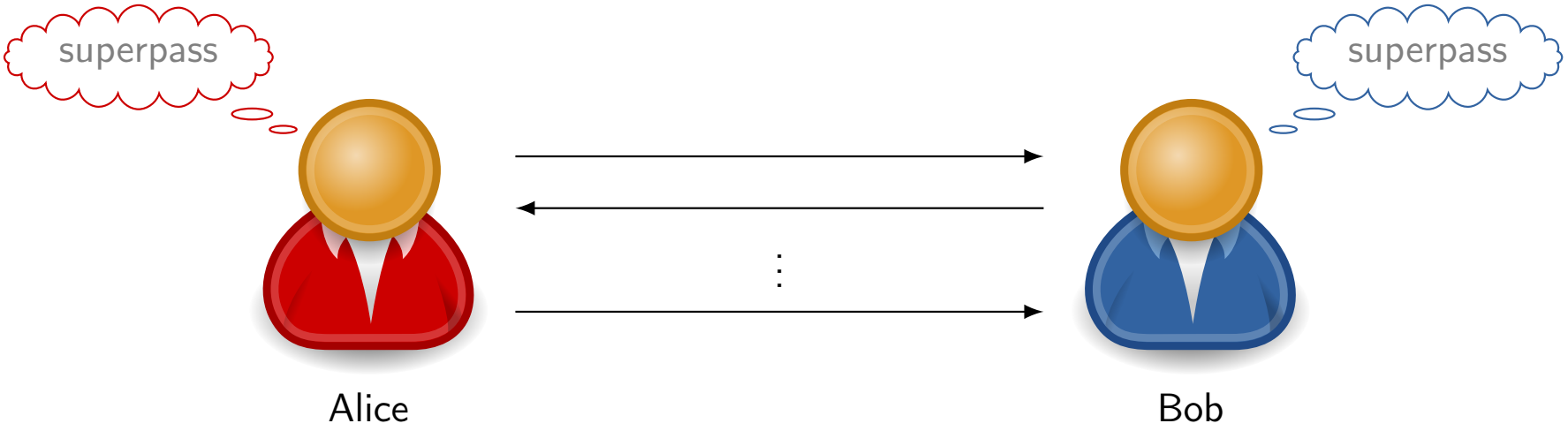
Michel Abdalla, *Fabrice Benhamouda*, Olivier Blazy,
Céline Chevalier, and David Pointcheval

École Normale Supérieure, CNRS and INRIA
Ruhr University Bochum
Université Panthéon-Assas

Asiacrypt 2013 — Bangalore, India
Monday, December 1

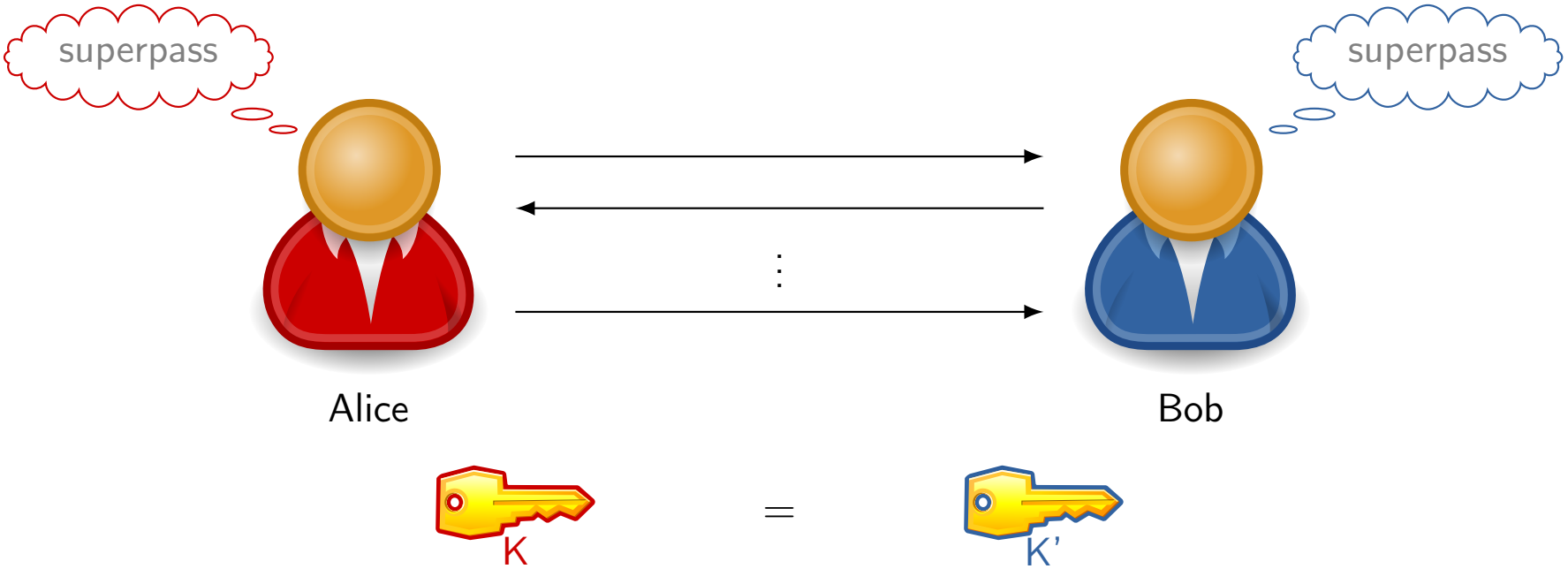
PAKE: Password-Authenticated Key Exchange

Goal: establishing a common secret key
from only a common low-entropy password



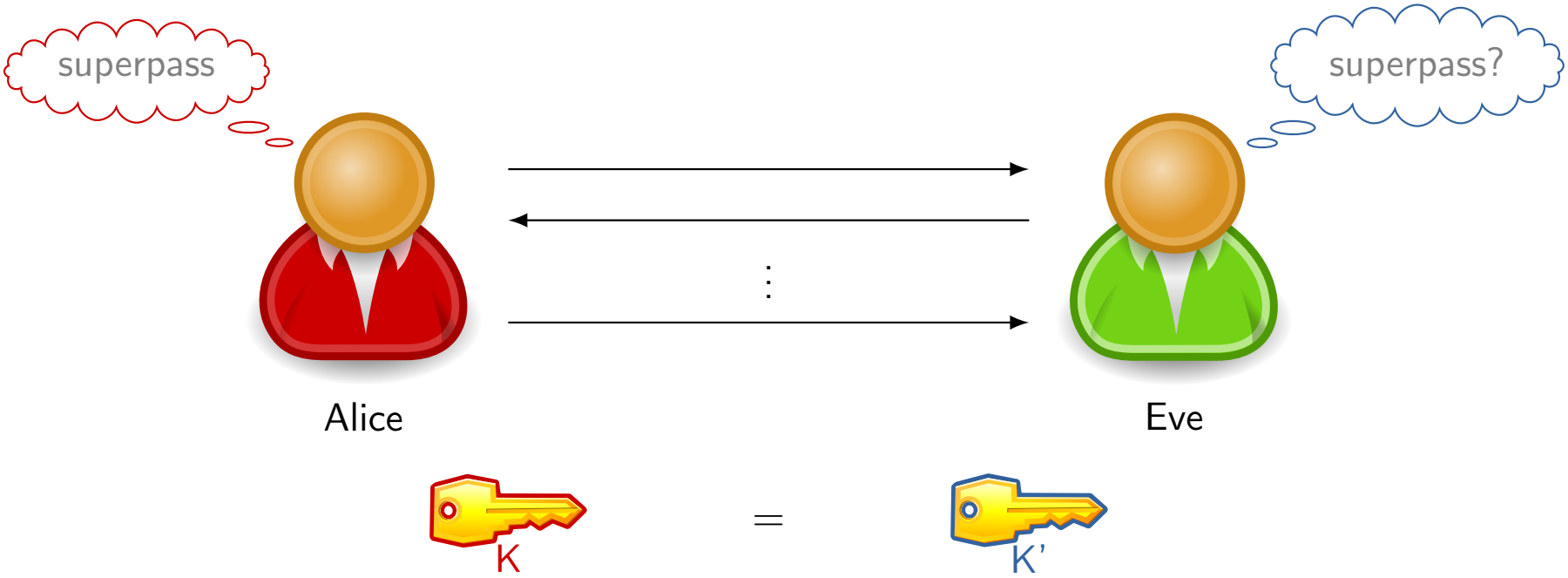
PAKE: Password-Authenticated Key Exchange

Goal: establishing a common secret key
from only a common low-entropy password



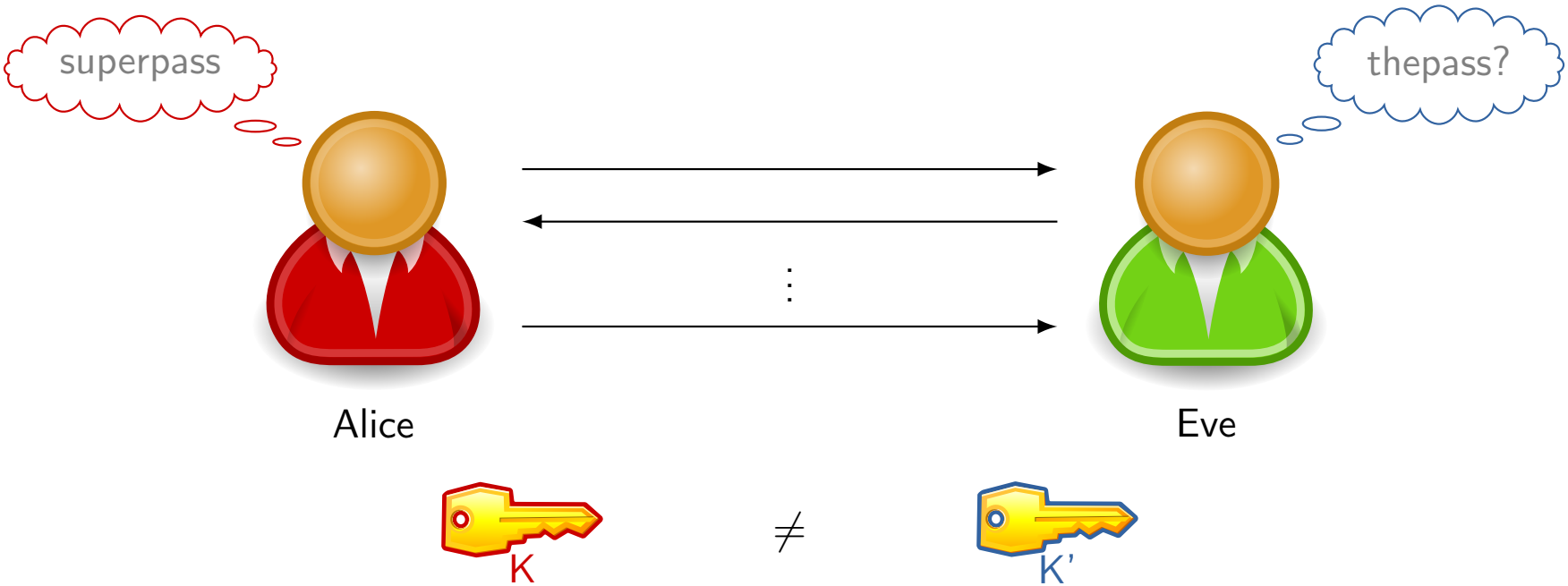
PAKE: Password-Authenticated Key Exchange

Goal: establishing a common secret key
from only a common low-entropy password



PAKE: Password-Authenticated Key Exchange

Goal: establishing a common secret key
from only a common low-entropy password



Intuitive security notion: only online dictionary attack works:

- at most one password can be tested per interaction;
- impossible to test password from an honest transcript.

PAKE: Password-Authenticated Key Exchange

Model Used

- one-round:
  - one flow per user (possibly simultaneous),
- UC [Can01],
- with adaptive corruptions (with erasures):
  - corruption of a user = learning the internal state,
  - possible at any time,
- in the standard model:
  - without random oracle.

UC PAKE: State of the Art

	Adaptive	One-round	Complexity (group elements)	Assumption
[BCLPR05]	✓	✗	very high	
[ACP09]	✓	✗	$\approx 44 \cdot m \cdot \mathfrak{K}$	DDH
[KV11]	✗	✓	≈ 140	DLIN
[BBCPV13]	✗	✓	≈ 22	SXDH

- m: size of the password
- $\mathfrak{K}$: security parameter

UC PAKE: State of the Art

	Adaptive	One-round	Complexity (group elements)	Assumption
[BCLPR05]	✓	✗	very high	
[ACP09]	✓	✗	$\approx 44 \cdot m \cdot \mathfrak{K}$	DDH
[KV11]	✗	✓	≈ 140	DLIN
[BBCPV13]	✗	✓	≈ 22	SXDH
here	✓	✓	$\approx 24 \cdot m$	SXDH

- m: size of the password
- $\mathfrak{K}$: security parameter

PAKE: Construction Sketch

In most efficient PAKE schemes:

- each user **commits** to his password, and
- using an **SPHF** (Smooth Projective Hash Function), they prove that they committed to the good password.

Construction introduced and used in [KOY01, GL03, KV11].

Non-Interactive Commitment

$\mathsf{Com}(\pi)$	generates a commitment C of π and a decommitment information δ
$\mathsf{VerCom}(C, \pi, \delta)$	checks C commits to π using δ

Non-Interactive Commitment

$\mathsf{Com}(\pi)$	generates a commitment C of π and a decommitment information δ
$\mathsf{VerCom}(C, \pi, \delta)$	checks C commits to π using δ

Non-Interactive Commitment

$\mathsf{Com}(\pi)$ generates a commitment C of π and a decommitment information δ

$\mathsf{VerCom}(C, \pi, \delta)$ checks C commits to π using δ

binding no poly-time adv. can find C, δ, δ' and $\pi \neq \pi'$ s.t.:

$$\mathsf{VerCom}(C, \pi, \delta) = 1 \quad \text{and} \quad \mathsf{VerCom}(C, \pi', \delta') = 1$$

hiding no poly-time adv. can distinguish:

$$\mathsf{Com}(\pi_0) \quad \text{and} \quad \mathsf{Com}(\pi_1)$$

for chosen π_0 and π_1.

Non-Interactive Commitment

$\mathsf{Com}(\pi)$ generates a commitment C of π and a decommitment information δ

$\mathsf{VerCom}(C, \pi, \delta)$ checks C commits to π using δ

binding no poly-time adv. can find C, δ, δ' and $\pi \neq \pi'$ s.t.:

$$\mathsf{VerCom}(C, \pi, \delta) = 1 \quad \text{and} \quad \mathsf{VerCom}(C, \pi', \delta') = 1$$

hiding no poly-time adv. can distinguish:

$$\mathsf{Com}(\pi_0) \quad \text{and} \quad \mathsf{Com}(\pi_1)$$

for chosen π_0 and π_1.

Implicit CRS: $\rho \xleftarrow{\$} \mathsf{SetupCom}(1^{\mathfrak{K}})$.

SPHF: Smooth Projective Hash Function [CS02, KV11]

NP language family $\mathrm{L}_{\mathsf{aux}} = \{C \in \mathcal{X} \mid \exists w,\ \mathcal{R}_{\mathsf{aux}}(C, w) = 1\}$
(w: witness)

$\mathsf{HashKG}(1^{\lambda})$	generates a hashing key hk
$\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C)$	computes the hash value H of $C \in \mathcal{X}$
$\mathsf{ProjKG}(\mathsf{hk})$	derives a projection key hp
$\mathsf{ProjHash}(\mathsf{hp}, \mathsf{aux}, C, w)$	computes the hash value H of $C \in \mathrm{L}_{\mathsf{aux}}$ (if $\mathcal{R}_{\mathsf{aux}}(C, w) = 1$)

In this talk:

hp does not depend on C (contrary to [GL03]) nor on aux.

SPHF: Smooth Projective Hash Function [CS02, KV11]

NP language family $\mathrm{L}_{\mathsf{aux}} = \{C \in \mathcal{X} \mid \exists w,\ \mathcal{R}_{\mathsf{aux}}(C, w) = 1\}$
(w: witness)

$\mathsf{HashKG}(1^{\mathfrak{K}})$	generates a hashing key hk
$\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C)$	computes the hash value H of $C \in \mathcal{X}$
$\mathsf{ProjKG}(\mathsf{hk})$	derives a projection key hp
$\mathsf{ProjHash}(\mathsf{hp}, \mathsf{aux}, C, w)$	computes the hash value H of $C \in \mathrm{L}_{\mathsf{aux}}$ (if $\mathcal{R}_{\mathsf{aux}}(C, w) = 1$)

In this talk:

hp does not depend on C (contrary to [GL03]) nor on aux.

SPHF: Smooth Projective Hash Function [CS02, KV11]

NP language family $\mathrm{L}_{\mathsf{aux}} = \{C \in \mathcal{X} \mid \exists w,\ \mathcal{R}_{\mathsf{aux}}(C, w) = 1\}$
(w: witness)

$\mathsf{HashKG}(1^{\mathfrak{K}})$	generates a hashing key hk
$\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C)$	computes the hash value H of $C \in \mathcal{X}$
$\mathsf{ProjKG}(\mathsf{hk})$	derives a projection key hp
$\mathsf{ProjHash}(\mathsf{hp}, \mathsf{aux}, C, w)$	computes the hash value H of $C \in \mathrm{L}_{\mathsf{aux}}$ (if $\mathcal{R}_{\mathsf{aux}}(C, w) = 1$)

In this talk:

hp does not depend on C (contrary to [GL03]) nor on aux.

SPHF: Smooth Projective Hash Function [CS02, KV11]

NP language family $\mathrm{L}_{\mathsf{aux}} = \{C \in \mathcal{X} \mid \exists w,\ \mathcal{R}_{\mathsf{aux}}(C, w) = 1\}$
(w: witness)

$\mathsf{HashKG}(1^{\mathfrak{K}})$	generates a hashing key hk
$\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C)$	computes the hash value H of $C \in \mathcal{X}$
$\mathsf{ProjKG}(\mathsf{hk})$	derives a projection key hp
$\mathsf{ProjHash}(\mathsf{hp}, \mathsf{aux}, C, w)$	computes the hash value H of $C \in \mathrm{L}_{\mathsf{aux}}$ (if $\mathcal{R}_{\mathsf{aux}}(C, w) = 1$)

In this talk:

hp does not depend on C (contrary to [GL03]) nor on aux.

SPHF: Smooth Projective Hash Function [CS02, KV11]

NP language family $\mathrm{L}_{\mathsf{aux}} = \{C \in \mathcal{X} \mid \exists w,\ \mathcal{R}_{\mathsf{aux}}(C, w) = 1\}$
(w: witness)

$\mathsf{HashKG}(1^{\kappa})$	generates a hashing key hk
$\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C)$	computes the hash value H of $C \in \mathcal{X}$
$\mathsf{ProjKG}(\mathsf{hk})$	derives a projection key hp
$\mathsf{ProjHash}(\mathsf{hp}, \mathsf{aux}, C, w)$	computes the hash value H of $C \in \mathrm{L}_{\mathsf{aux}}$ (if $\mathcal{R}_{\mathsf{aux}}(C, w) = 1$)

In this talk:

hp does not depend on C (contrary to [GL03]) nor on aux.

Properties of SPHF

correctness for any hk and corresponding hp, for all $C \in \mathrm{L}_{\mathsf{aux}}$ and w such that $\mathcal{R}_{\mathsf{aux}}(C, w) = 1$:

$$\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C) = \mathsf{ProjHash}(\mathsf{hp}, \mathsf{aux}, C, w);$$

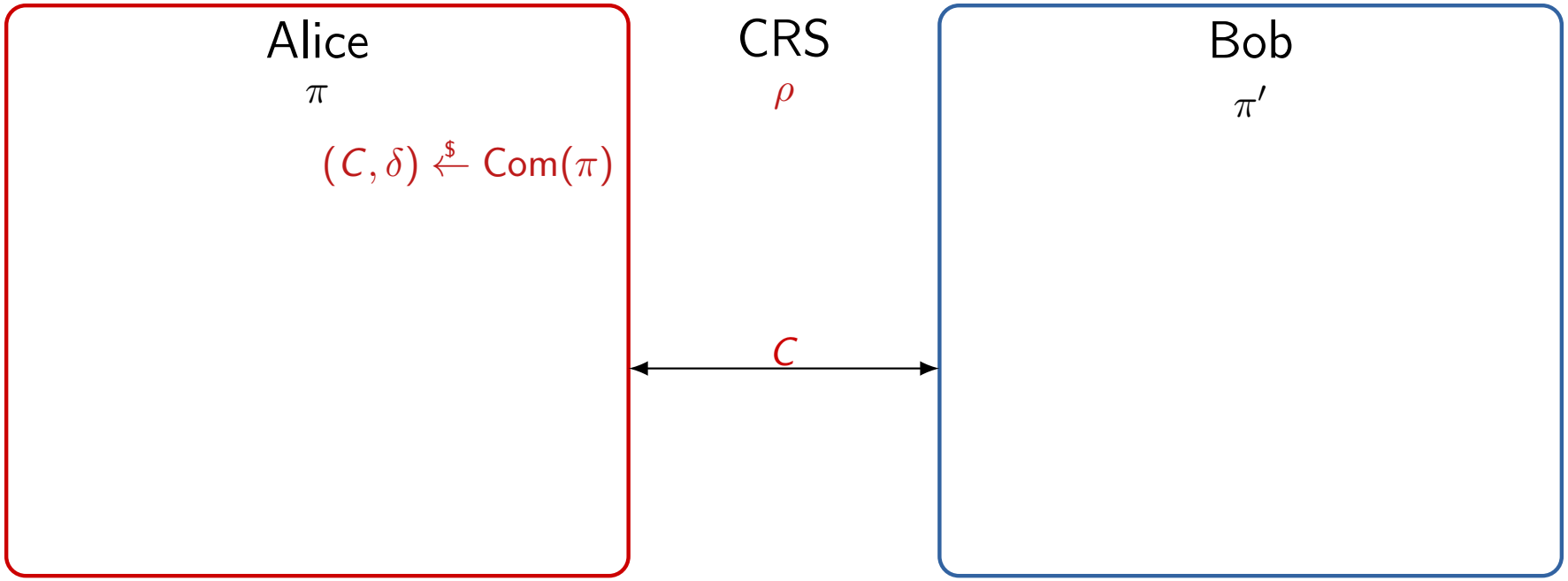
smoothness (definition of [KV11])

- for any function f onto $\mathcal{X} \setminus \mathrm{L}_{\mathsf{aux}}$,
- given a projection key hp,
- $C = f(\mathsf{hp}) \notin \mathrm{L}_{\mathsf{aux}}$,
- $\mathsf{Hash}(\mathsf{hk}, \mathsf{aux}, C) \approx_s$ random.

Contributions

- formalization of SPHF-friendly commitments:
  - ◇ implicit in [ACP09];
- construction of an efficient SPHF-friendly commitment:
  - ◇ inspired by [CF01, CLOS02, ACP09];
  - \+ $O(m)$ elements instead of $O(m\mathfrak{K})$ elements;
- applications:
  - adaptive UC commitment;
  - first one-round adaptive UC PAKE;
  - 1-out-of-k UC OT more efficient than [CKWZ13].

PAKE Construction Sketch

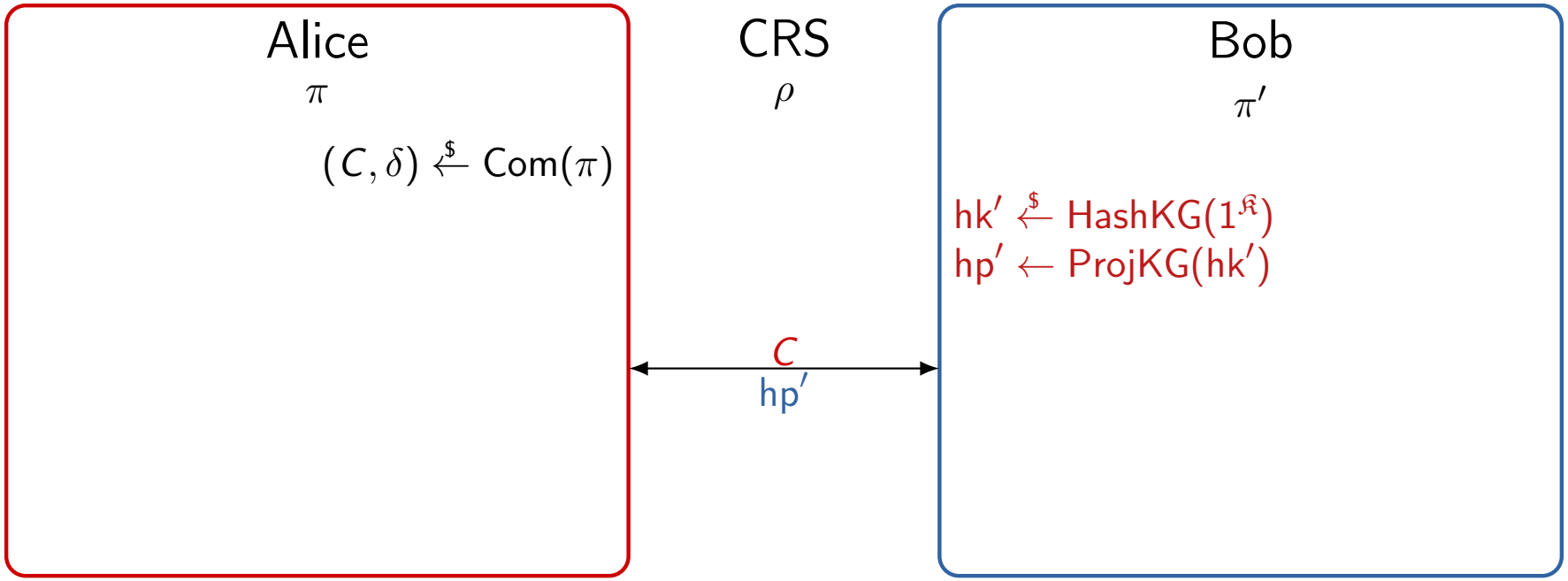


Language for SPHF: valid commitments of aux ($= \pi$ or π'):

$$\mathcal{R}_{\mathsf{aux}}(C, \delta) = 1 \quad \Longleftrightarrow \quad \mathsf{VerCom}(C, \mathsf{aux}, \delta) = 1.$$

Correctness if $\pi = \pi'$, $H_A = H'_A$;

PAKE Construction Sketch

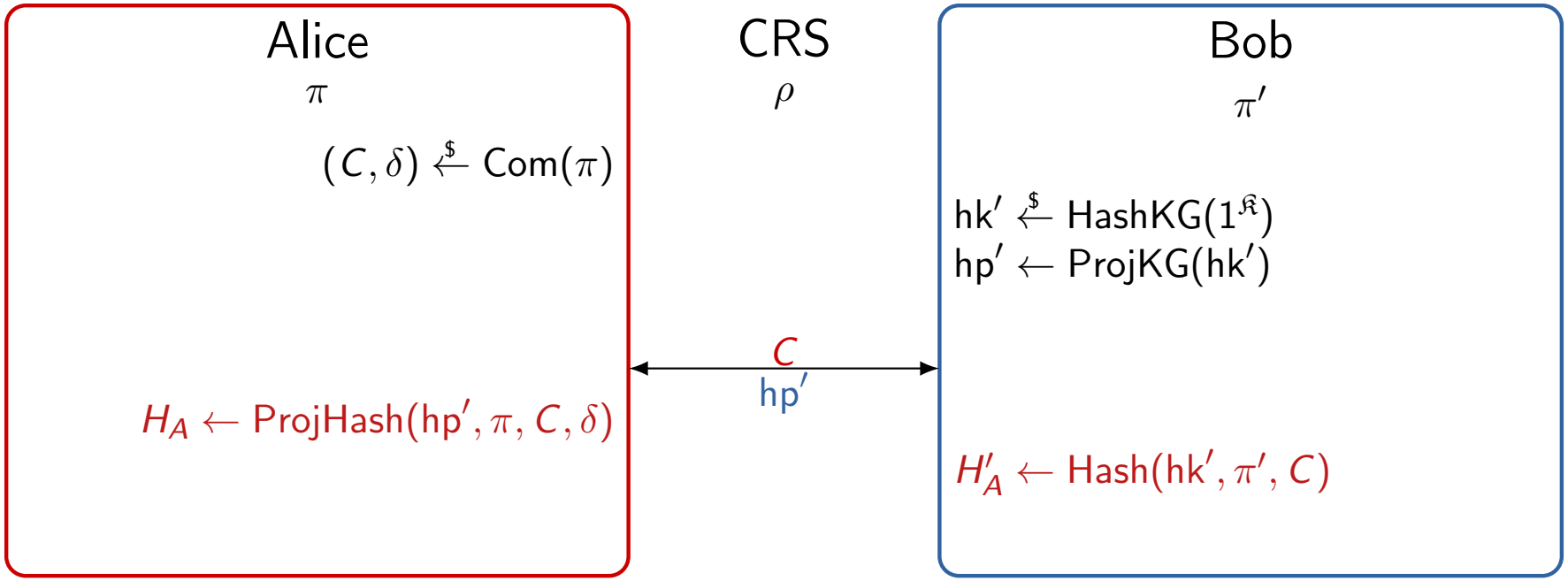


Language for SPHF: valid commitments of aux ($= \pi$ or π'):

$$\mathcal{R}_{\mathsf{aux}}(C, \delta) = 1 \iff \mathsf{VerCom}(C, \mathsf{aux}, \delta) = 1.$$

Correctness if $\pi = \pi'$, $H_A = H'_A$;

PAKE Construction Sketch

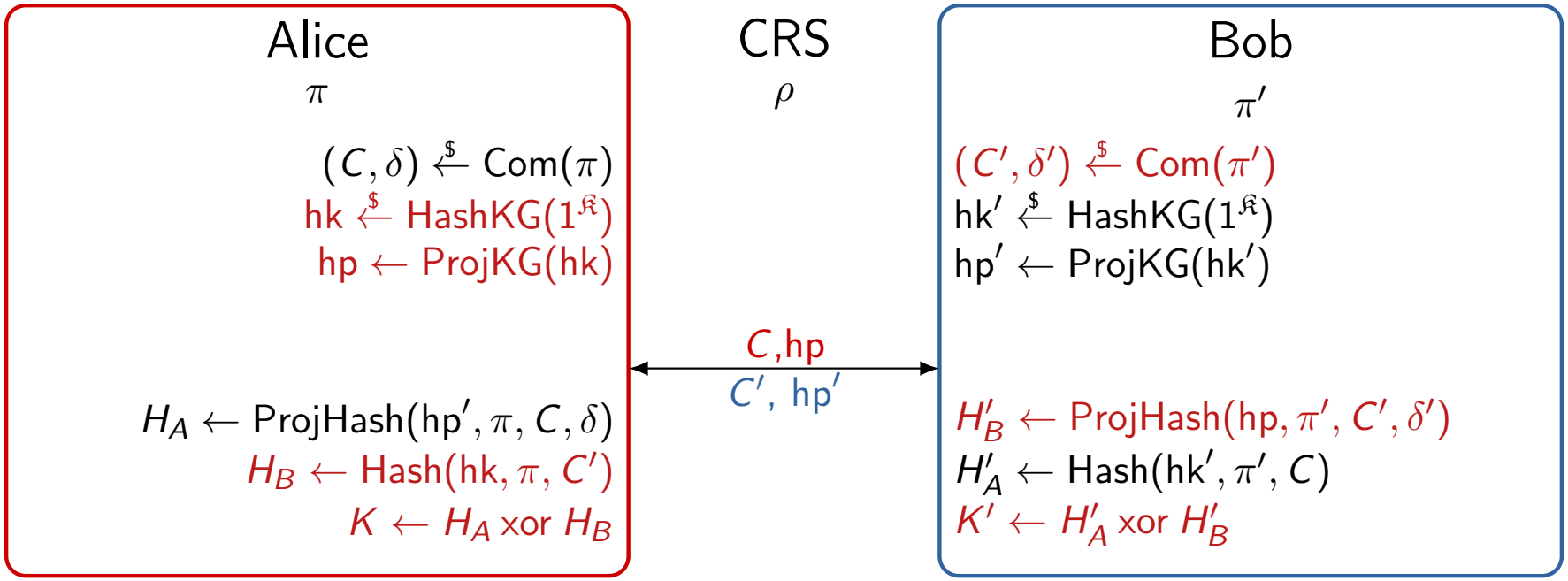


Language for SPHF: valid commitments of aux ($= \pi$ or π'):

$$\mathcal{R}_{\mathsf{aux}}(C, \delta) = 1 \quad \Longleftrightarrow \quad \mathsf{VerCom}(C, \mathsf{aux}, \delta) = 1.$$

Correctness if $\pi = \pi'$, $H_A = H'_A$;

PAKE Construction Sketch

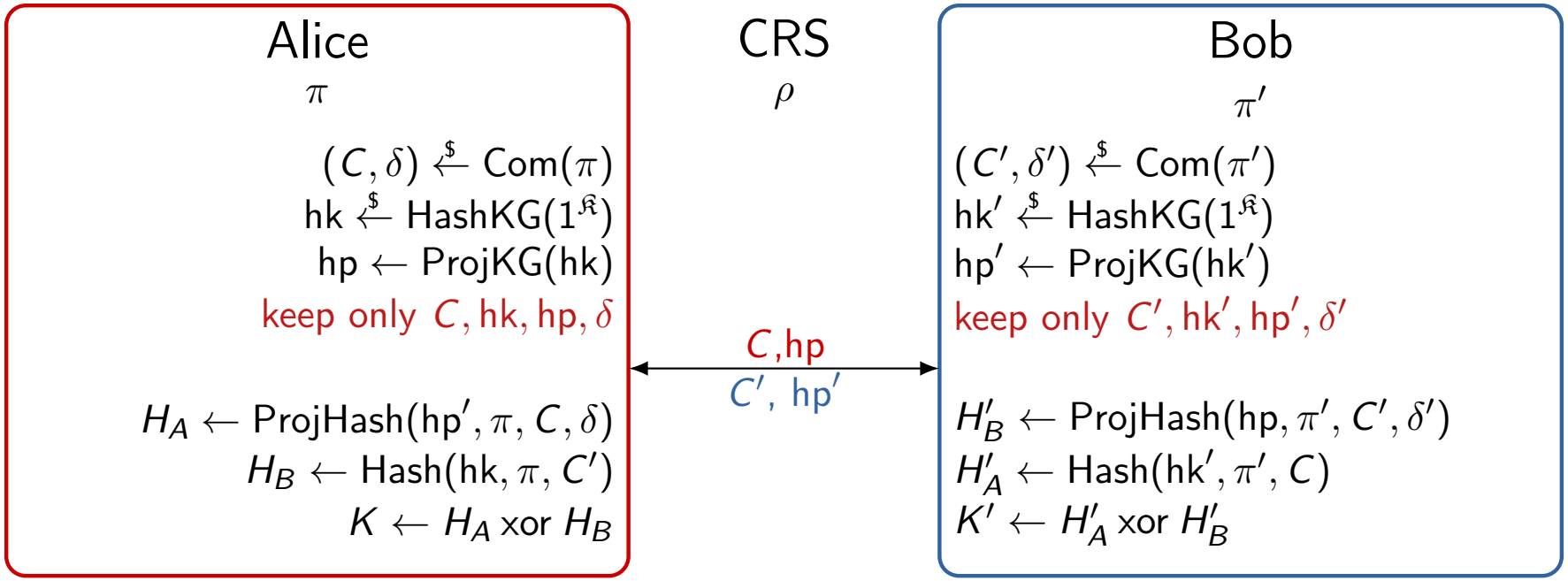


Language for SPHF: valid commitments of aux ($= \pi$ or π'):

$$\mathcal{R}_{\mathsf{aux}}(C, \delta) = 1 \quad \Longleftrightarrow \quad \mathrm{VerCom}(C, \mathsf{aux}, \delta) = 1.$$

Correctness if $\pi = \pi'$, $H_A = H'_A$, $H_B = H'_B$ and $K = K'$;

PAKE Construction Sketch



Language for SPHF: valid commitments of aux ($= \pi$ or π'):

$$\mathcal{R}_{\mathsf{aux}}(C, \delta) = 1 \quad \Longleftrightarrow \quad \mathsf{VerCom}(C, \mathsf{aux}, \delta) = 1.$$

Correctness if $\pi = \pi'$, $H_A = H'_A$, $H_B = H'_B$ and $K = K'$;

Security ?
Equivocability

In UC model, with adaptive corruptions:

- need to simulate a user w/o knowing its password π;
  - $\rightarrow$ need to generate hp and C
- and when corrupted, we learn π
  - $\rightarrow$ need to generate δ:

$$\mathsf{VerCom}(C, \pi, \delta) = 1.$$

Security ?
Equivocability

In UC model, with adaptive corruptions:

- need to simulate a user w/o knowing its password π;
  - $\rightarrow$ need to generate hp and $(C, \mathsf{eqk}) \xleftarrow{\$} \mathsf{SimCom}(\tau)$
- and when corrupted, we learn π
  - $\rightarrow$ need to generate $\delta \leftarrow \mathsf{OpenCom}(\mathsf{eqk}, \pi)$:

$$\mathsf{VerCom}(C, \pi, \delta) = 1.$$

$\longrightarrow$ commitment property: **equivocablility**

Security ?
Equivocability

In UC model, with adaptive corruptions:

- need to simulate a user w/o knowing its password π;
  - $\rightarrow$ need to generate hp and $(C, \mathsf{eqk}) \xleftarrow{\$} \mathsf{SimCom}(\tau)$
- and when corrupted, we learn π
  - $\rightarrow$ need to generate $\delta \leftarrow \mathsf{OpenCom}(\mathsf{eqk}, \pi)$:

$$\mathsf{VerCom}(C, \pi, \delta) = 1.$$

$\longrightarrow$ commitment property: **equivocablility**

$$\text{hiding} \quad \Leftarrow \quad \text{equivocability}$$

Security ?

Equivocable Commitments — Examples

[Ped91] Pedersen scheme:

CRS	cyclic group $\mathbb{G}$, generators g and $T = g^t$
$\mathsf{Com}(\pi)$	$r \xleftarrow{\$} \mathbb{Z}_p$, $C = g^r \cdot T^\pi$, $\delta = r$
$\mathsf{VerCom}(C, \pi, \delta)$	$C \stackrel{?}{=} g^\delta \cdot T^\pi$
$\mathsf{SimCom}(t)$	$u \xleftarrow{\$} \mathbb{Z}_p$, $C = g^r$, $\mathsf{eqk} = u$
$\mathsf{OpenCom}(\mathsf{eqk}, \pi)$	$\delta = \mathsf{eqk} - tx$

Security ?

Equivocable Commitments — Examples

[Ped91] Pedersen scheme:

CRS	cyclic group $\mathbb{G}$, generators g and $T = g^t$
$\mathsf{Com}(\pi)$	$r \xleftarrow{\$} \mathbb{Z}_p$, $C = g^r \cdot T^\pi$, $\delta = r$
$\mathsf{VerCom}(C, \pi, \delta)$	$C \stackrel{?}{=} g^\delta \cdot T^\pi$
$\mathsf{SimCom}(t)$	$u \xleftarrow{\$} \mathbb{Z}_p$, $C = g^r$, $\mathsf{eqk} = u$
$\mathsf{OpenCom}(\mathsf{eqk}, \pi)$	$\delta = \mathsf{eqk} - tx$

Security ?

Equivocable Commitments — Examples

[Ped91] Pedersen scheme:

CRS	cyclic group $\mathbb{G}$, generators g and $T = g^t$
$\mathrm{Com}(\pi)$	$r \xleftarrow{\$} \mathbb{Z}_p$, $C = g^r \cdot T^\pi$, $\delta = r$
$\mathrm{VerCom}(C, \pi, \delta)$	$C \stackrel{?}{=} g^\delta \cdot T^\pi$
$\mathrm{SimCom}(t)$	$u \xleftarrow{\$} \mathbb{Z}_p$, $C = g^r$, $\mathrm{eqk} = u$
$\mathrm{OpenCom}(\mathrm{eqk}, \pi)$	$\delta = \mathrm{eqk} - tx$

Security ?

Equivocable Commitments — Examples

[Ped91] Pedersen scheme:

$$\begin{aligned}
\text{CRS} &\quad \text{cyclic group } \mathbb{G}, \text{ generators } g \text{ and } T = g^t \\
\text{Com}(\pi) &\quad r \xleftarrow{\$} \mathbb{Z}_p,\ C = g^r \cdot T^\pi,\ \delta = r \\
\text{VerCom}(C, \pi, \delta) &\quad C \stackrel{?}{=} g^\delta \cdot T^\pi \\
\text{SimCom}(t) &\quad u \xleftarrow{\$} \mathbb{Z}_p,\ C = g^r,\ \text{eqk} = u \\
\text{OpenCom}(\text{eqk}, \pi) &\quad \delta = \text{eqk} - tx
\end{aligned}$$

[Har11] Haralambiev TC4 scheme:

$$\begin{aligned}
\text{CRS} &\quad (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2),\ T = g_2^t \\
\text{Com}(\pi) &\quad r \xleftarrow{\$} \mathbb{Z}_p,\ C = g_2^r \cdot T^\pi,\ \delta = g_1^r \\
\text{VerCom}(C, \pi, \delta) &\quad e(g_1, C/T^\pi) \stackrel{?}{=} e(\delta, g_2) \\
\text{SimCom}(t) &\quad u \xleftarrow{\$} \mathbb{Z}_p,\ C = g_2^r,\ \text{eqk} = u \\
\text{OpenCom}(\text{eqk}, \pi) &\quad \delta = g_1^{\text{eqk}-tx}
\end{aligned}$$

Security ?

Extractability / Strong Extractibility

In UC model:

- need to check if the adv. committed to a valid password:
  - $\rightarrow$ need to extract committed value

Security ?

Extractability / Strong Extractibility

In UC model:

- need to check if the adv. committed to a valid password:

  $\rightarrow$ need to extract committed value

  i.e., no poly-time adv. can find C, δ, π s.t.:

$$\mathsf{VerCom}(C, \pi, \delta) = 1 \quad \text{and} \quad \mathsf{ExtCom}(\tau, C) \neq \pi$$

$\longrightarrow$ commitment property: **extractability**

$$\text{binding} \quad \Leftarrow \quad \text{extractability}$$

Security ?

Extractability / Strong Extractibility

In UC model:

- need to check if the adv. committed to a valid password:
  - $\rightarrow$ need to extract committed value

  i.e., no poly-time adv. can find C, δ, π s.t.:

$$\mathsf{VerCom}(C, \pi, \delta) = 1 \quad \text{and} \quad \mathsf{ExtCom}(\tau, C) \neq \pi$$

- even when simulating commitments !

$\longrightarrow$ commitment property: **strong extractability**

$$\text{binding} \quad \Leftarrow \quad \text{extractability} \quad \Leftarrow \quad \text{strong extractibility}$$

Security ?

Extractable Commitments — Examples

ElGamal [ElG84] or Cramer-Shoup [CS98] encryption scheme

CRS	cyclic group $\mathbb{G}_1$, public key pk
$\mathsf{Com}(\pi)$	$r \xleftarrow{\$} \mathbb{Z}_p$, $C \leftarrow \mathsf{CS}(\mathsf{pk}, \pi; r)$, $\delta = r$
$\mathsf{VerCom}(C, \pi, \delta)$	$C \stackrel{?}{=} \mathsf{CS}(\mathsf{pk}, \pi; \delta)$
$\mathsf{ExtCom}(\mathsf{sk}, C)$	$\mathsf{Dec}(\mathsf{sk}, C)$

Security ?

Extractable Commitments — Examples

ElGamal [ElG84] or Cramer-Shoup [CS98] encryption scheme

CRS	cyclic group $\mathbb{G}_1$, public key pk
$\mathsf{Com}(\pi)$	$r \xleftarrow{\$} \mathbb{Z}_p$, $C \leftarrow \mathsf{CS}(\mathsf{pk}, \pi; r)$, $\delta = r$
$\mathsf{VerCom}(C, \pi, \delta)$	$C \stackrel{?}{=} \mathsf{CS}(\mathsf{pk}, \pi; \delta)$
$\mathsf{ExtCom}(\mathsf{sk}, C)$	$\mathsf{Dec}(\mathsf{sk}, C)$

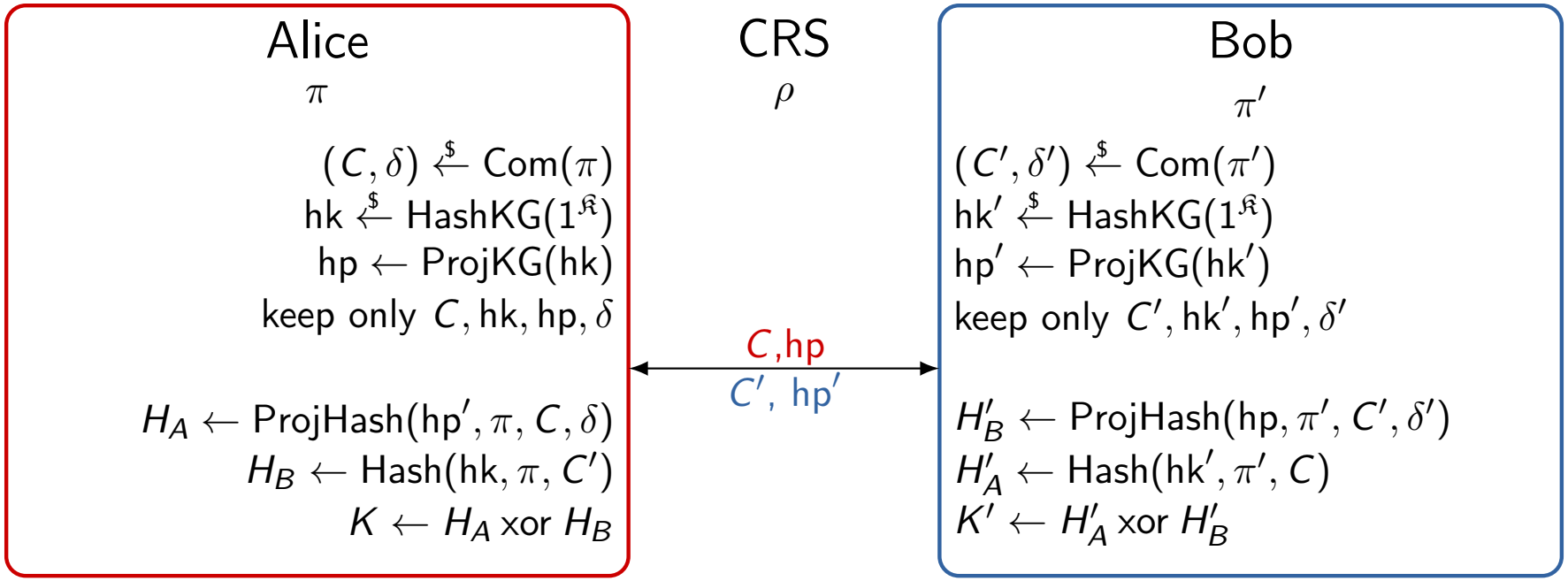
Security ?

Extractable Commitments — Examples

ElGamal [ElG84] or Cramer-Shoup [CS98] encryption scheme

CRS	cyclic group $\mathbb{G}_1$, public key pk
$\mathsf{Com}(\pi)$	$r \xleftarrow{\$} \mathbb{Z}_p$, $C \leftarrow \mathsf{CS}(\mathsf{pk}, \pi; r)$, $\delta = r$
$\mathsf{VerCom}(C, \pi, \delta)$	$C \stackrel{?}{=} \mathsf{CS}(\mathsf{pk}, \pi; \delta)$
$\mathsf{ExtCom}(\mathsf{sk}, C)$	$\mathsf{Dec}(\mathsf{sk}, C)$

PAKE Construction Sketch



Language for SPHF: valid commitments of aux ($= \pi$ or π'):

$$\mathcal{R}_{\mathsf{aux}}(C, \delta) = 1 \quad \Longleftrightarrow \quad \mathsf{VerCom}(C, \mathsf{aux}, \delta) = 1.$$

Correctness if $\pi = \pi'$, $H_A = H'_A$, $H_B = H'_B$ and $K = K'$;

Security ?
Robustness

$$L_\pi = \{C \in \mathcal{X} \mid \exists \delta,\ \mathsf{VerCom}(C, \pi, \delta) = 1\}.$$

For a strong extractable commitment, we may have:

$$L_\pi = \{C \in \mathcal{X} \mid \exists \delta, \pi',\ \mathsf{VerCom}(C, \pi', \delta) = 1\} = \mathcal{X}.$$

Security ?
Robustness

$$L_\pi = \{C \in \mathcal{X} \mid \exists \delta,\ \mathsf{VerCom}(C, \pi, \delta) = 1\}.$$

For a strong extractable commitment, we may have:

$$L_\pi = \{C \in \mathcal{X} \mid \exists \delta, \pi',\ \mathsf{VerCom}(C, \pi', \delta) = 1\} = \mathcal{X}.$$

$\longrightarrow$ commitment property: **robustness**:

no poly-time adv. can find C s.t.:

$$\exists \delta, \pi,\ \mathsf{VerCom}(C, \pi, \delta) = 1 \quad \text{and} \quad \mathsf{ExtCom}(\tau, C) \neq \pi.$$

$$\text{strong extractibility} \quad \Leftarrow \quad \text{robustness}$$

Summary

$$\begin{array}{lllllll} \text{hiding} & \Leftarrow & \text{equivocability} & \Leftarrow & \text{strong equivocability} & & \\ \text{binding} & \Leftarrow & \text{extractability} & \Leftarrow & \text{strong extractibility} & \Leftarrow & \text{robustness} \end{array}$$

Summary

hiding $\Leftarrow$ equivocability $\Leftarrow$ strong equivocability

binding $\Leftarrow$ extractability $\Leftarrow$ strong extractibility $\Leftarrow$ robustness

equivocability + robustness $\rightsquigarrow$ SPHF-friendly commitment

Summary

hiding $\Leftarrow$ equivocability $\Leftarrow$ strong equivocability

binding $\Leftarrow$ extractability $\Leftarrow$ strong extractibility $\Leftarrow$ robustness

equivocability + robustness $\rightsquigarrow$ SPHF-friendly commitment

equivocability + strong extractability $\rightsquigarrow$ (adaptive) UC commitment

Summary

hiding $\Leftarrow$ equivocability $\Leftarrow$ strong equivocability
binding $\Leftarrow$ extractability $\Leftarrow$ strong extractibility $\Leftarrow$ robustness

equivocability + robustness $\rightsquigarrow$ SPHF-friendly commitment

equivocability + strong extractability $\rightsquigarrow$ (adaptive) UC commitment
strong equivocability + extractability $\rightsquigarrow$ (adaptive) UC commitment

State of the Art

	SPHF	C	δ	Assumpt.
[ACP09]	✓	$(m + 16m\mathfrak{K}) \times \mathbb{G}$	$2m\mathfrak{K} \times \mathbb{Z}_p$	DDH
[FLM11], 1	✗	$5 \times \mathbb{G}$	$16 \times \mathbb{G}$	DLIN
[FLM11], 2	✗	$37 \times \mathbb{G}$	$3 \times \mathbb{G}$	DLIN
here	✓	$8m \times \mathbb{G}_1 + m \times \mathbb{G}_2$	$m \times \mathbb{Z}_p$	SXDH

State of the Art

	SPHF	C	δ	Assumpt.
[ACP09]	✓	$(m + 16m\mathfrak{K}) \times \mathbb{G}$	$2m\mathfrak{K} \times \mathbb{Z}_p$	DDH
[FLM11], 1	✗	$5 \times \mathbb{G}$	$16 \times \mathbb{G}$	DLIN
[FLM11], 2	✗	$37 \times \mathbb{G}$	$3 \times \mathbb{G}$	DLIN
here	✓	$8m \times \mathbb{G}_1 + m \times \mathbb{G}_2$	$m \times \mathbb{Z}_p$	SXDH

Why schemes in [FLM11] are not robust ?

- C is an encryption of π;
- δ is a NIZK that C encrypts π
  - $\rightarrow$ can be simulated!

Our SPHF-Friendly Commitment Scheme

- bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$,
- $T = g_2^t$ for Haralambiev TC4 commitment, $\pi = 101$
- pk for Cramer-Shoup in $\mathbb{G}_1$.

1	0	1
r_1	r_2	r_3
$d_{1,1} = g_1^{r_1}$	$d_{2,0} = g_1^{r_2}$	$d_{3,1} = g_1^{r_3}$
$a_1 = g_2^{r_1} \cdot T^1$	$a_2 = g_2^{r_2} \cdot T^0$	$a_3 = g_3^{r_3} \cdot T^1$

$C = (a_1, a_2, a_3)$ $\delta = (d_1, d_2, d_3)$

Our SPHF-Friendly Commitment Scheme

- bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$,
- $T = g_2^t$ for Haralambiev TC4 commitment, $\pi = 101$
- pk for Cramer-Shoup in $\mathbb{G}_1$.

1	0	1
r_1	r_2	r_3
$d_{1,1} = g_1^{r_1}$	$d_{2,0} = g_1^{r_2}$	$d_{3,1} = g_1^{r_3}$
$a_1 = g_2^{r_1} \cdot T^1$	$a_2 = g_2^{r_2} \cdot T^0$	$a_3 = g_3^{r_3} \cdot T^1$
$b_{1,1} = \mathsf{CS}(d_{1,1}; s_{1,1})$	$b_{2,0} = \mathsf{CS}(d_{2,0}; s_{2,0})$	$b_{3,1} = \mathsf{CS}(d_{3,1}; s_{3,1})$

$C = (a_1, a_2, a_3, b_{1,1}, b_{2,0}, b_{3,1})$ $\delta = (s_{1,1}, s_{2,0}, s_{3,1})$

Our SPHF-Friendly Commitment Scheme

- bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$,
- $T = g_2^t$ for Haralambiev TC4 commitment, $\pi = 101$
- pk for Cramer-Shoup in $\mathbb{G}_1$.

1	0	1
r_1	r_2	r_3
$d_{1,0} = 1$ $d_{1,1} = g_1^{r_1}$	$d_{2,0} = g_1^{r_2}$ $d_{2,1} = 1$	$d_{3,0} = 1$ $d_{3,1} = g_1^{r_3}$
$a_1 = g_2^{r_1} \cdot T^1$	$a_2 = g_2^{r_2} \cdot T^0$	$a_3 = g_3^{r_3} \cdot T^1$
$b_{1,0} = \mathsf{CS}(d_{1,0}; s_{1,0})$ $b_{1,1} = \mathsf{CS}(d_{1,1}; s_{1,1})$	$b_{2,0} = \mathsf{CS}(d_{2,0}; s_{2,0})$ $b_{2,1} = \mathsf{CS}(d_{2,1}; s_{2,1})$	$b_{3,0} = \mathsf{CS}(d_{3,0}; s_{3,0})$ $b_{3,1} = \mathsf{CS}(d_{3,1}; s_{3,1})$

$$C = (a_1, a_2, a_3, b_{1,0}, b_{1,1}, b_{2,0}, b_{2,1}, b_{3,0}, b_{3,1}) \qquad \delta = (s_{1,1}, s_{2,0}, s_{3,1})$$

Our SPHF-Friendly Commitment Scheme

- bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$,
- $T = g_2^t$ for Haralambiev TC4 commitment, $\pi = 101$
- pk for Cramer-Shoup in $\mathbb{G}_1$.

1	0	1
r_1	r_2	r_3
$d_{1,0} = g_1^{r1+t}$ $d_{1,1} = g_1^{r_1}$	$d_{2,0} = g_1^{r_2}$ $d_{2,1} = g_1^{r2-t}$	$d_{3,0} = g_1^{r3+t}$ $d_{3,1} = g_1^{r_3}$
$a_1 = g_2^{r_1} \cdot T^1$	$a_2 = g_2^{r_2} \cdot T^0$	$a_3 = g_3^{r_3} \cdot T^1$
$b_{1,0} = \mathsf{CS}(d_{1,0}; s_{1,0})$ $b_{1,1} = \mathsf{CS}(d_{1,1}; s_{1,1})$	$b_{2,0} = \mathsf{CS}(d_{2,0}; s_{2,0})$ $b_{2,1} = \mathsf{CS}(d_{2,1}; s_{2,1})$	$b_{3,0} = \mathsf{CS}(d_{3,0}; s_{3,0})$ $b_{3,1} = \mathsf{CS}(d_{3,1}; s_{3,1})$

equivocability ?

$C = (a_1, a_2, a_3, b_{1,0}, b_{1,1}, b_{2,0}, b_{2,1}, b_{3,0}, b_{3,1})$ $\delta = (s_{1,1}, s_{2,0}, s_{3,1})$

Our SPHF-Friendly Commitment Scheme

- bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$,
- $T = g_2^t$ for Haralambiev TC4 commitment, $\pi = 101$
- pk for Cramer-Shoup in $\mathbb{G}_1$.

1	0	1
r_1	r_2	r_3
$d_{1,0} = 1$ $d_{1,1} = g_1^{r_1}$	$d_{2,0} = g_1^{r_2}$ $d_{2,1} = 1$	$d_{3,0} = 1$ $d_{3,1} = g_1^{r_3}$
$a_1 = g_2^{r_1} \cdot T^1$	$a_2 = g_2^{r_2} \cdot T^0$	$a_3 = g_3^{r_3} \cdot T^1$
$b_{1,0} = \mathsf{CS}(d_{1,0}; s_{1,0})$ $b_{1,1} = \mathsf{CS}(d_{1,1}; s_{1,1})$	$b_{2,0} = \mathsf{CS}(d_{2,0}; s_{2,0})$ $b_{2,1} = \mathsf{CS}(d_{2,1}; s_{2,1})$	$b_{3,0} = \mathsf{CS}(d_{3,0}; s_{3,0})$ $b_{3,1} = \mathsf{CS}(d_{3,1}; s_{3,1})$

robustness ?

$C = (a_1, a_2, a_3, b_{1,0}, b_{1,1}, b_{2,0}, b_{2,1}, b_{3,0}, b_{3,1})$ $\delta = (s_{1,1}, s_{2,0}, s_{3,1})$

Our SPHF-Friendly Commitment Scheme

The SPHF

- language: pairing equations over Cramer-Shoup ciphertexts;
- SPHF: using methods in [BBCPV13].

Thank you for your attention!

- formalization of SPHF-friendly commitments:
  - ◇ implicit in [ACP09];
- construction of an efficient SPHF-friendly commitment:
  - ◇ inspired by [CF01, CLOS02, ACP09];
  - \+ $O(m)$ elements instead of $O(m\mathfrak{K})$ elements;
- applications:
  - adaptive UC commitment;
  - first one-round adaptive UC PAKE;
  - 1-out-of-k UC OT more efficient than [CKWZ13].

References I

Michel Abdalla, Céline Chevalier, and David Pointcheval.

Smooth projective hashing for conditionally extractable commitments.

In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 671–689. Springer, August 2009.

Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud.

New techniques for SPHFs and efficient one-round PAKE protocols.

In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 449–475. Springer, August 2013.

References II

Boaz Barak, Ran Canetti, Yehuda Lindell, Rafael Pass, and Tal Rabin.

Secure computation without authentication.

In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 361–377. Springer, August 2005.

Ran Canetti.

Universally composable security: A new paradigm for cryptographic protocols.

In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

Ran Canetti and Marc Fischlin.

Universally composable commitments.

In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40. Springer, August 2001.

References III

Seung Geol Choi, Jonathan Katz, Hoeteck Wee, and Hong-Sheng Zhou.

Efficient, adaptively secure, and composable oblivious transfer with a single, global CRS.

In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 73–88. Springer, February / March 2013.

Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai.

Universally composable two-party and multi-party secure computation.

In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

References IV

Ronald Cramer and Victor Shoup.

A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack.

In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 13–25. Springer, August 1998.

Ronald Cramer and Victor Shoup.

Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption.

In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, April / May 2002.

References V

Taher ElGamal.

A public key cryptosystem and a signature scheme based on discrete logarithms.

In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, August 1984.

Marc Fischlin, Benoît Libert, and Mark Manulis.

Non-interactive and re-usable universally composable string commitments with adaptive security.

In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 468–485. Springer, December 2011.

References VI

Rosario Gennaro and Yehuda Lindell.

A framework for password-based authenticated key exchange.

In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 524–543. Springer, May 2003.

`http://eprint.iacr.org/2003/032.ps.gz`.

Kristiyan Haralambiev.

Efficient Cryptographic Primitives for Non-Interactive Zero-Knowledge Proofs and Applications.

PhD thesis, New York University, 2011.

Jonathan Katz, Rafail Ostrovsky, and Moti Yung.

Efficient password-authenticated key exchange using human-memorable passwords.

In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 475–494. Springer, May 2001.

References VII

Jonathan Katz and Vinod Vaikuntanathan.

Round-optimal password-based authenticated key exchange.

In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 293–310. Springer, March 2011.

Torben P. Pedersen.

Non-interactive and information-theoretic secure verifiable secret sharing.

In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, August 1991.