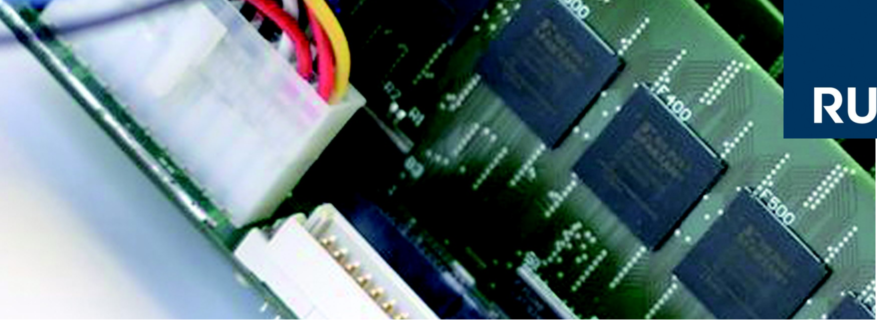


RUHR-UNIVERSITÄT BOCHUM

Efficient UC-Secure Authenticated Key-Exchange for Algebraic Languages

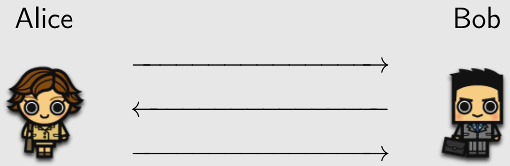
PKC 2013,

Fabrice Ben Hamouda **Olivier Blazy** Céline Chevalier
David Pointcheval Damien Vergnaud
Horst Görtz Institute for IT Security / Ruhr-University Bochum
ENS / CNRS / INRIA / Université Panthéon-Assas

Horst Görtz Institut
für IT-Sicherheit

Outline

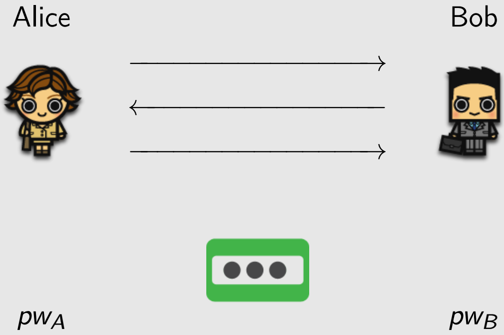
Authenticated Key Exchange



K_{AB}

Share a common session key iff everything goes well.

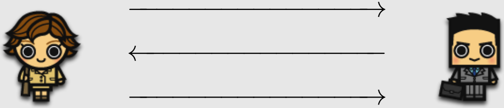
Password Authenticated Key Exchange [BM92]



Share a common session key iff they possess the same password.

Secret Handshakes [BDSS03]

Alice Bob



σ_A σ_B

Share a common session key iff their signatures fit.

Credential Authenticated Key Exchange [CCGS10]

Alice Bob

$Cred(A)$ $Cred(B)$

Share a common session key iff they possess the required credentials.

Language Authenticated Key Exchange

Alice Bob

w_A w_B

Share a common session key iff their (words/languages) fit.

RUB

Outline

Cramer Shoup Encryption

Definition [CS02]

- Setup(1^λ): Generates a multiplicative group $(p, \mathbb{G}, g_1, g_2)$.
- EKeyGen$_\mathcal{E}$(param): dk $= (\mu_{1,2}, \nu_{1,2}, \eta_{1,2}) \stackrel{\$}{\leftarrow} \mathbb{Z}_p^6$,
  pk $= (c = g_1^{\mu_1} g_2^{\mu_2}, d = g_1^{\nu_1} g_2^{\nu_2}, h = g_1^{\eta_1} g_2^{\eta_2})$.
- Encrypt(pk, $M; \alpha$): For M, and $\alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, defines $\mathcal{C} = CS(M; \alpha)$ as
  $\left(u = (g_1^\alpha, g_2^\alpha), e = Mh^\alpha, v = (cd^\xi)^\alpha\right)$.
  $\xi = \mathsf{Hash}(u, e)$
- Decrypt(dk $= (\mu, \nu, \eta), \mathcal{C} = (u, e, v)$):
  If $v = \prod u_i^{\mu_i + \xi\nu_i}$, then $M = e \cdot \prod u_i^{-\eta_i}$.

IND-CCA under DDH

Double Cramer Shoup Encryption

Definition

- Setup(1^λ): Generates a multiplicative group $(p, \mathbb{G}, g_1, g_2)$.
- EKeyGen$_\mathcal{E}$(param): dk $\stackrel{\$}{\leftarrow} \mathbb{Z}_p^6$, pk.
- Encrypt$_1$(pk, $M; \alpha$): $\mathcal{C} = CS(M; \alpha)$.
- Encrypt$_2$(pk, $N, \xi; \alpha'$): For N, and $\alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, defines $\mathcal{C}' = CS'(N, \xi; \alpha)$ as
  $\left(u' = (g_1^{\alpha'}, g_2^{\alpha'}), e' = Mh^{\alpha'}, v' = (cd^\xi)^{\alpha'}\right)$.
- Decrypt(dk $= (\mu, \nu, \eta), \mathcal{C} = (u, e, v), \mathcal{C}'$):
  If $v = \prod u_i^{\mu_i + \xi\nu_i}$, then $M = e \cdot \prod u_i^{-\eta_i}$.
  If $v' = \prod u_i'^{\mu_i + \xi\nu_i}$, then $N = e' \cdot \prod u_i'^{-\eta_i}$.

IND-PD-CCA under DDH (IND-CCA on CS, IND-CPA on CS')

Multi Double Cramer Shoup Encryption

Definition

- Setup(1^λ): Generates a multiplicative group $(p, \mathbb{G}, g_1, g_2)$.
- EKeyGen$_\mathcal{E}$(param): dk $\stackrel{\$}{\leftarrow} \mathbb{Z}_p^6$, pk.
- Encrypt$_1$(pk, $\mathbf{M}; \boldsymbol{\alpha}$): $\mathcal{C} = CS(\mathbf{M}; \boldsymbol{\alpha})$, where $\xi = \mathrm{Hash}(\mathbf{u}, \mathbf{e})$.
- Encrypt$_2$(pk, $\mathbf{N}, \xi; \boldsymbol{\alpha}'$): $\mathcal{C}' = CS'(\mathbf{N}, \xi; \boldsymbol{\alpha}')$.
- Decrypt(dk $= (\mu, \nu, \eta), \mathcal{C}, \mathcal{C}'$):
  If $\mathbf{v} = \prod \mathbf{u_i}^{\mu_i + \xi\nu_i}$, then $\mathbf{M} = \mathbf{e} \cdot \prod \mathbf{u_i}^{-\eta_i}$.
  If $\mathbf{v}' = \prod \mathbf{u}_\mathbf{i}'^{\mu_i + \xi\nu_i}$, then $\mathbf{N} = \mathbf{e}' \cdot \prod \mathbf{u}_\mathbf{i}'^{-\eta_i}$.

IND-PD-CCA under DDH.

Smooth Projective Hash Functions

Definition [CS02,GL03]

Let $\{H\}$ be a family of functions:

- X, domain of these functions
- L, subset (a language) of this domain

such that, for any point x in L, $H(x)$ can be computed by using

- either a *secret* hashing key hk: $H(x) = \text{Hash}_L(\text{hk}; x)$;
- or a *public* projected key hp: $H'(x) = \text{ProjHash}_L(\text{hp}; x, w)$

Public mapping $\text{hk} \mapsto \text{hp} = \text{ProjKG}_L(\text{hk}, x)$

Properties

For any $x \in X$, $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$
For any $x \in L$, $H(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$ $\quad$ w witness that $x \in L$

Properties

For any $x \in X$, $H(x) = \mathrm{Hash}_L(\mathrm{hk}; x)$
For any $x \in L$, $H(x) = \mathrm{ProjHash}_L(\mathrm{hp}; x, w)$ $\quad w$ witness that $x \in L$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Properties

For any $x \in X$, $H(x) = \mathsf{Hash}_L(\mathsf{hk}; x)$
For any $x \in L$, $H(x) = \mathsf{ProjHash}_L(\mathsf{hp}; x, w)$ $\quad w$ witness that $x \in L$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Pseudo-Randomness

For any $x \in L$, $H(x)$ is pseudo-random, without a witness w

Properties

For any $x \in X$, $H(x) = \text{Hash}_L(\text{hk}; x)$
For any $x \in L$, $H(x) = \text{ProjHash}_L(\text{hp}; x, w)$ $\quad w$ witness that $x \in L$

Smoothness

For any $x \notin L$, $H(x)$ and hp are independent

Pseudo-Randomness

For any $x \in L$, $H(x)$ is pseudo-random, without a witness w

The latter property requires L to be a hard-partitioned subset of X:

Hard-Partitioned Subset

L is a hard-partitioned subset of X if it is computationally hard to distinguish a random element in L from a random element in $X \setminus L$

Straightforward Languages

- Diffie Hellman / Linear Tuple

$(g, h, G = g^a, H = h^a)$ Valid Diffie Hellman tuple?

hp : $g^\kappa h^\lambda$ $\quad$ $\text{hp}^a = G^\kappa H^\lambda$

Oblivious Transfer, Implicit Opening of a ciphertext

Straightforward Languages

- Diffie Hellman / Linear Tuple

$(g, h, G = g^a, H = h^a)$ Valid Diffie Hellman tuple?

hp : $g^\kappa h^\lambda$ $\quad$ $\text{hp}^a = G^\kappa H^\lambda$

Oblivious Transfer, Implicit Opening of a ciphertext

$(U = u^a, V = v^b, W = g^{a+b})$ Valid Linear tuple?

hp : $u^\kappa g^\lambda, v^\mu g^\lambda$ $\quad$ $\text{hp}_1^a \text{hp}_2^b = U^\kappa V^\mu W^\lambda$

Straightforward Languages

- Diffie Hellman / Linear Tuple
- Conjunction / Disjunction

$\mathcal{L}_1 \cap \mathcal{L}_2$
$\mathsf{hp} : \mathsf{hp}_1, \mathsf{hp}_2$
$\wedge A_i$

Simultaneous verification
$H'_1 \cdot H'_2 = H_1 \cdot H_2$

Straightforward Languages

- Diffie Hellman / Linear Tuple
- Conjunction / Disjunction

$\mathcal{L}_1 \cup \mathcal{L}_2$

$\mathsf{hp} = \mathsf{hp}_1, \mathsf{hp}_2, \mathsf{hp}_\Delta$

Is it a bit?

One out of 2 conditions

$H' = \mathcal{L}_1 ? \mathsf{hp}_1^{w_1} : \mathsf{hp}_2^{w_2} \cdot \mathsf{hp}_\Delta = X_1^{\mathsf{hk}_1}$

Advanced Languages

- (Linear) Cramer-Shoup Encryption

$(u_1 = g_1^r, u_2 = g_2^r, e = h^r M, v = (cd^\xi)^r)$
hp : $g_1^\kappa g_2^\mu (cd^\xi)^\eta h^\lambda$

Verifiability of the CS
$\text{hp}^r = u_1^\kappa u_2^\mu v^\eta (e/M)^\lambda$

Implicit Opening of a ciphertext, verifiability of a ciphertext, PAKE

Advanced Languages

- (Linear) Cramer-Shoup Encryption

$(u_1 = g_1^r, u_2 = g_2^r, e = h^r M, v = (cd^\xi)^r)$

hp : $g_1^\kappa g_2^\mu (cd^\xi)^\eta h^\lambda$

Verifiability of the CS

$\mathrm{hp}^r = u_1^\kappa u_2^\mu v^\eta (e/M)^\lambda$

Implicit Opening of a ciphertext, verifiability of a ciphertext, PAKE

$(g_1^r, g_2^s, g_3^{r+s}, h_1^r h_2^s M, (c_1 d_1^\xi)^r (c_2 d_2^\xi)^s)$

hp : $g_1^\kappa g_3^\theta (c_1 d_1^\xi)^\eta h^\lambda, g_2^\mu g_3^\theta (c_2 d_2^\xi)^\eta h^\lambda$

Verifiability of the LCS

$\mathrm{hp}_1^r \mathrm{hp}_2^s = u_1^\kappa u_2^\mu u_3^\theta v^\eta (e/M)^\lambda$

Advanced Languages

- (Linear) Cramer-Shoup Encryption
- Commitment of a commitment

$(U = u^a, V = v^s, G = h^s g^a)$ ELin

hp : $u^\eta g^\lambda, v^\theta h^\lambda$ $\quad \mathrm{hp}_1^a \mathrm{hp}_2^s = U^\eta V^\theta G^\lambda$

Advanced Languages

- (Linear) Cramer-Shoup Encryption
- Commitment of a commitment
- Linear Pairing Equations

$$\left(\prod_{i \in A_k} e(\mathcal{Y}_i, \mathcal{A}_{k,i})\right) \cdot \left(\prod_{i \in B_k} \mathcal{Z}_i^{\mathfrak{z}_{k,i}}\right) = \mathcal{D}_k$$

For each variables: $\text{hp}_i : u^{\kappa_i} g^{\lambda}, v^{\mu_i} g^{\lambda}$

$$\left(\prod_{i \in A_k} e(\text{hp}_i^{w_i}, \mathcal{A}_{k,i})\right) \cdot \left(\prod_{i \in B_k} \text{HP}_i^{\mathfrak{z}_{k,i} w_i}\right) =$$

$$\left(\prod_{i \in A_k} e(H_i, \mathcal{A}_{k,i})\right) \cdot \left(\prod_{i \in B_k} H_i^{\mathfrak{z}_{k,i}}\right) / \mathcal{D}_k^{\lambda}$$

Knowledge of a secret key, Knowledge of a (secret) signature on a (secret) message valid under a (secret) verification key, ...

Commitment à la Lindell [Lin11]

Alice		Bob
$\mathcal{C}, \mathcal{C}' = DCS(\mathbf{M}, \mathbf{1}; \boldsymbol{\alpha}),$		
$\pi = Ped(\mathcal{C}', t, \mathbf{M})$	$\xrightarrow{\mathcal{C}, \pi}$	$\boldsymbol{\epsilon} \xleftarrow{\$} \mathbb{Z}_p^n,$
	$\xleftarrow{\boldsymbol{\epsilon}, \mathsf{hp}}$	$\mathsf{hp}_i = g_1^{\mu_i} g_2^{\nu_i} h^{\lambda_i} (cd^{\xi})^{\theta_i}$
$\boldsymbol{z} = \epsilon \boldsymbol{\alpha_1} + \boldsymbol{\alpha_2}$	$\xrightarrow{t, \mathcal{C}'}$	

Commitment à la Lindell

[Lin11]

Alice		Bob
$\mathcal{C}, \mathcal{C}' = DCS(\mathbf{M}, \mathbf{1}; \boldsymbol{\alpha})$,		
$\pi = Ped(\mathcal{C}', t, \mathbf{M})$	$\xrightarrow{\mathcal{C}, \pi}$	$\boldsymbol{\epsilon} \stackrel{\$}{\leftarrow} \mathbb{Z}_p^n$,
	$\xleftarrow{\boldsymbol{\epsilon}, \mathsf{hp}}$	$\mathsf{hp}_i = g_1^{\mu_i} g_2^{\nu_i} h^{\lambda_i} (cd^{\xi})^{\theta_i}$
$\boldsymbol{z} = \epsilon \boldsymbol{\alpha}_1 + \boldsymbol{\alpha}_2$	$\xrightarrow{t, \mathcal{C}'}$	
	$\xrightarrow{\mathsf{hp}^{\boldsymbol{z}}, \boldsymbol{M}}$	$\mathsf{Hash}(\mathcal{C}^{\boldsymbol{\epsilon}} \mathcal{C}', \boldsymbol{M}, \mathsf{hk})$

- Self-Randomizable Language

- Self-Randomizable Language
- Double-Step PD-CCA Commitment

- Self-Randomizable Language
- Double-Step PD-CCA Commitment
- Implicit Decommitment

RUB

Outline

Language Authenticated Key Exchange

Alice		Bob

$$\xrightarrow{\mathcal{C}(\mathcal{L}_B, \mathcal{L}'_A, M_B), \pi(\mathcal{C}')}$$

$$\xleftarrow{\mathcal{C}(\mathcal{L}'_B, \mathcal{L}_A, M_A), \mathsf{hp}_B, \epsilon}$$

$$\xrightarrow{\mathsf{hp}_A, \mathcal{C}'(1,1,1)}$$

$H_B \cdot H'_A$ | | $H'_B \cdot H_A$

Same value iff languages are as expected, and users know witnesses.

Secret Handshakes *for the same secret signing authority*

Alice

Bob

$$\xrightarrow{\mathcal{C}(\mathcal{L}(\sigma, \mathsf{vk}_A, id_B), \mathcal{L}(\sigma, \mathsf{vk}_A, id_A), \sigma(A)), \pi(\mathcal{C}')}$$

$$\xleftarrow{\mathcal{C}(\mathcal{L}(\sigma, \mathsf{vk}_B, id_B), \mathcal{L}(\sigma, \mathsf{vk}_B, id_A), \sigma(B)), \mathsf{hp}_B, \epsilon}$$

$$\xrightarrow{\mathsf{hp}_A, \mathcal{C}'(1,1,1)}$$

$H_B \cdot H'_A$

$H'_B \cdot H_A$

Ciphertext of a Waters Signature valid under the committed vk:

$$e(\sigma_1, g) = e(h, \mathsf{vk}) \cdot e(id_*, \sigma_2)$$

Password Authenticated Key Exchange

Alice		Bob
	$\xrightarrow{\mathcal{C}(pw_B),\ \pi(\mathcal{C}')}$	
	$\xleftarrow{\mathcal{C}(pw_A),\ \mathsf{hp}_B,\ \epsilon}$
	$\xrightarrow{\mathsf{hp}_A,\ \mathcal{C}'(1)}$

$H_B \cdot H'_A$ $\qquad$ $H'_B \cdot H_A$

Share a common session key iff they possess the same password.

Password Authenticated Key Exchange

Alice

Bob

$$u^{r_A}, v^{r_A}, pw_B h^{r_A}, (cd^{\xi_A})^{r_A} \longrightarrow$$

$$g^t k^{\mathrm{Hash}(\mathcal{C}'_A)}$$

$$\longleftarrow pw_A h^{r_B}, g^{r_B}$$

$$\mathrm{hp}_B : u^{\lambda_B} v^{\mu_B} h^{\eta_B} (cd^{\xi_A})^{\theta_B}, \epsilon$$

$$\mathcal{C}'_A = (u^{s_A}, v^{s_A}, h^{s_A}, (cd^{\xi_A})^{s_A}) \longrightarrow$$

$$t, \mathrm{hp}_A : g^{\lambda_A} h^{\eta_A}$$

$$\mathcal{C}_{B,-pw_A}^{\mathrm{hk}_A} \cdot \mathrm{hp}_B^{s_A + \epsilon r_A}$$

$$\mathrm{hp}_A^{r_B} \cdot \mathcal{C}^{*}_{A,-pw_B}{}^{\mathrm{hk}_B}$$

Outline

Extensions and Open Questions

✓ We presented a general Framework to instantiate several AKE protocols.

Extensions and Open Questions

- ✓ We presented a general Framework to instantiate several AKE protocols.
- ✓ This allows to produce efficient UC instantiations under classical assumptions (DDH,DLin,...)

Extensions and Open Questions

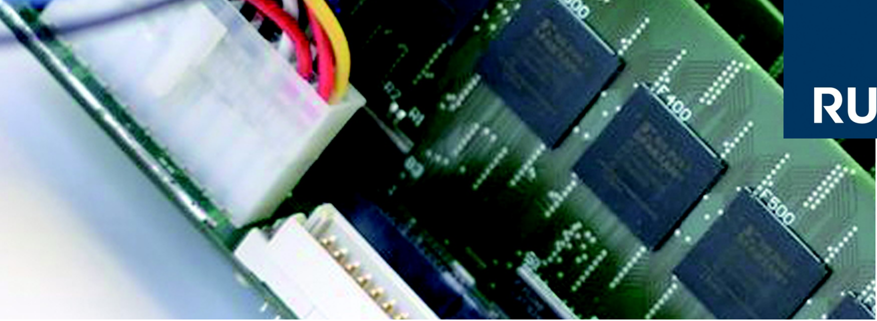
- ✓ We presented a general Framework to instantiate several AKE protocols.
- ✓ This allows to produce efficient UC instantiations under classical assumptions (DDH,DLin,...)
- ✓ Concrete examples for PAKE, v-PAKE, several Secret Handshakes, CAKE, . . .

Extensions and Open Questions

- ✓ We presented a general Framework to instantiate several AKE protocols.
- ✓ This allows to produce efficient UC instantiations under classical assumptions (DDH,DLin,...)
- ✓ Concrete examples for PAKE, v-PAKE, several Secret Handshakes, CAKE, . . .
- ✓ New manageable languages with SPHF implicit proofs of knowledge

Extensions and Open Questions

- ✓ We presented a general Framework to instantiate several AKE protocols.
- ✓ This allows to produce efficient UC instantiations under classical assumptions (DDH,DLin,...)
- ✓ Concrete examples for PAKE, v-PAKE, several Secret Handshakes, CAKE, . . .
- ✓ New manageable languages with SPHF implicit proofs of knowledge
- ✓ Several new tools: multi-commitment on CS, revisited commitment à la Lindell, . . .



RUHR-UNIVERSITÄT BOCHUM

Many thanks for your attention!

Any questions?

More details are available in the full version...

Horst Görtz Institut
für IT-Sicherheit