Anonymous Credential Schemes with Encrypted Attributes

Bart Mennink (K.U.Leuven)

joint work with

Jorge Guajardo (Philips Research)

Berry Schoenmakers (TU Eindhoven)

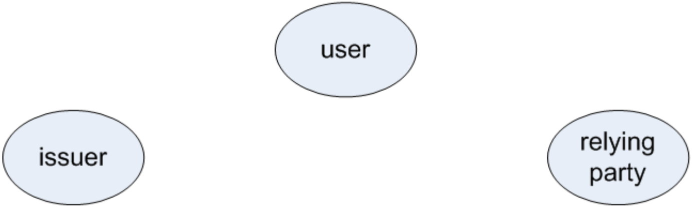
Conference on Cryptology And Network Security

Kuala Lumpur, Malaysia

14 December 2010

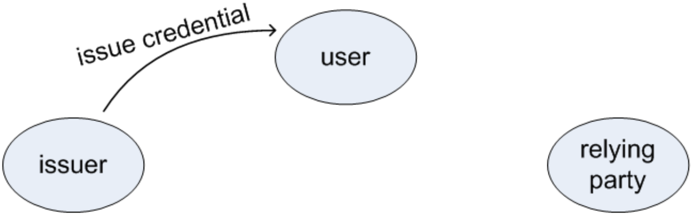
Outline

Anonymous Credential Schemes



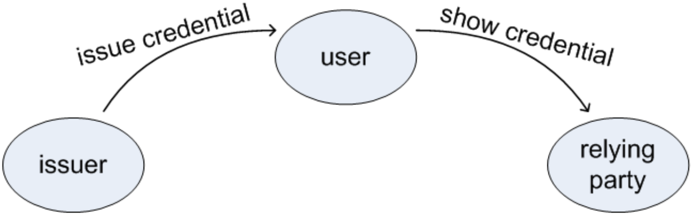
- 3 types of parties: *issuers, users, relying parties (verifiers)*

Anonymous Credential Schemes



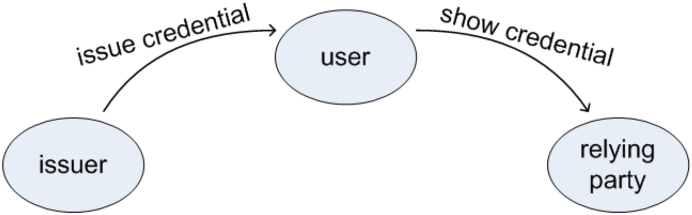
- 3 types of parties: *issuers, users, relying parties (verifiers)*
- Issuer issues credential (on some attributes) to user

Anonymous Credential Schemes



- 3 types of parties: *issuers, users, relying parties (verifiers)*
- Issuer issues credential (on some attributes) to user
- User shows credential to a relying party

Anonymous Credential Schemes



- 3 types of parties: *issuers, users, relying parties (verifiers)*
- Issuer issues credential (on some attributes) to user
- User shows credential to a relying party
- Required security properties: unforgeability and unlinkability
- Examples: digital passports, identity management, ...

Anonymous Credential Schemes

Theory:

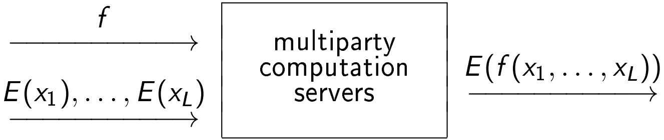
- Introduced by Chaum in 1982-1986
- Several efficient constructions, most prominently by
  - Brands [Bra93, Bra95, Bra99]
  - Camenisch-Lysyanskaya [CL01, CL02, CL04]
- Plus variations and extensions

Practice:

- eCash, DigiCash (Chaum)
- CAFE project
- Idemix (IBM, Camenisch-Lysyanskaya credentials)
- U-Prove (Microsoft, Brands credentials)

Secure Function Evaluation (SFE)

- A number of parties jointly and securely compute a function f on secret data



Secure Function Evaluation (SFE)

- A number of parties jointly and securely compute a function f on secret data

$$\begin{array}{c} \xrightarrow{\quad f \quad} \\ \xrightarrow{E(x_1), \ldots, E(x_L)} \end{array} \boxed{\begin{array}{c}\text{multiparty}\\ \text{computation}\\ \text{servers}\end{array}} \xrightarrow{E(f(x_1, \ldots, x_L))}$$

Yao's millionaires protocol (1982)

- Alice and Bob want to compare their wealth

Secure Function Evaluation (SFE)

- A number of parties jointly and securely compute a function f on secret data

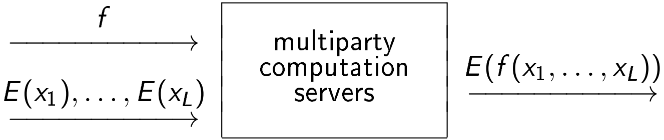
$$\begin{array}{c} \xrightarrow{\quad f \quad} \\ \xrightarrow{E(x_1), \ldots, E(x_L)} \end{array} \boxed{\begin{array}{c}\text{multiparty}\\ \text{computation}\\ \text{servers}\end{array}} \xrightarrow{E(f(x_1, \ldots, x_L))}$$

Yao's millionaires protocol (1982)

- Alice and Bob want to compare their wealth
- Both encrypt their fortunes: $E(x_A), E(x_B)$

Secure Function Evaluation (SFE)

- A number of parties jointly and securely compute a function f on secret data



Yao's millionaires protocol (1982)

- Alice and Bob want to compare their wealth
- Both encrypt their fortunes: $E(x_A), E(x_B)$
- Jointly execute SFE protocol to compute $x_A < x_B$
- Security: the protocol should not leak any other info about x_A, x_B

Secure Function Evaluation (SFE)

Theory:

- Introduced by Yao in 1982-1986
- Main approaches to multiparty computation by
  - Ben-Or *et al.* [BGW87]
  - Goldreich *et al.* [GMW87]
  - Cramer *et al.* [CDN01]

Practice:

- E-voting, e-auctions, ...
- Fairplay (2004), VIFF (2007), Sharemind (2008), and more
- Commercial activity: new company Partisia

Combining Credential Schemes and SFE

Problem:

- Input to SFE should be correct or meaningful
  - → One can employ credentials to guarantee this
  - → But SFE servers are not allowed to learn the attributes, while standard credential schemes require the user to know these

Combining Credential Schemes and SFE

Problem:

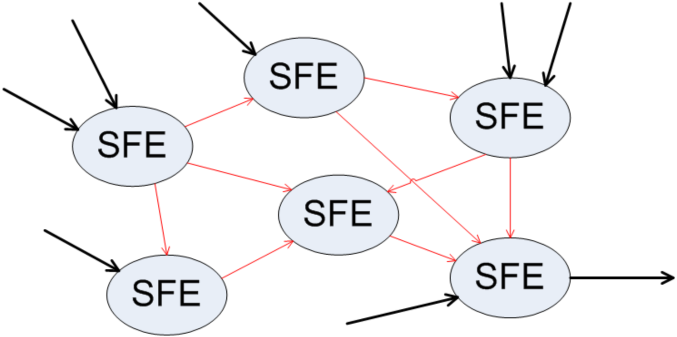
- Input to SFE should be correct or meaningful
  - → One can employ credentials to guarantee this
  - → But SFE servers are not allowed to learn the attributes, while standard credential schemes require the user to know these

Solution:

Anonymous Credentials on Encrypted Attributes

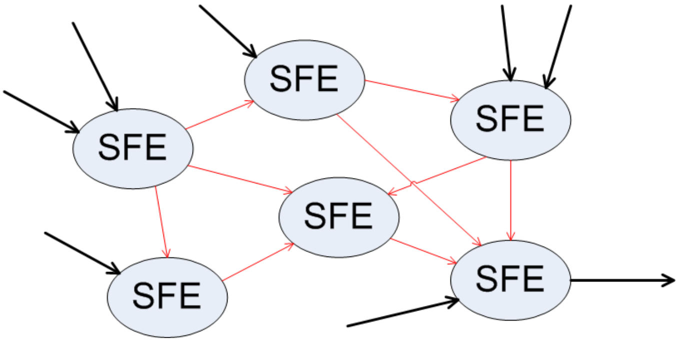
Combining Credential Schemes and SFE

- Lead to networks of SFEs with anonymous links connecting inputs and outputs



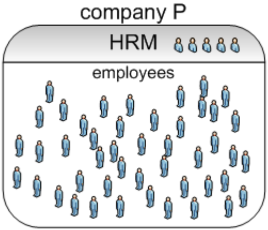
Combining Credential Schemes and SFE

- Lead to networks of SFEs with anonymous links connecting inputs and outputs



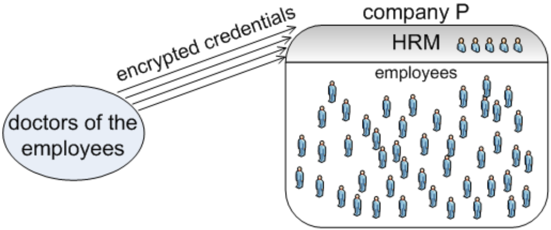
- In general, anonymous credentials with encrypted attributes can be used if the user is not allowed or does not want to know the attributes

Medical Data of Employees



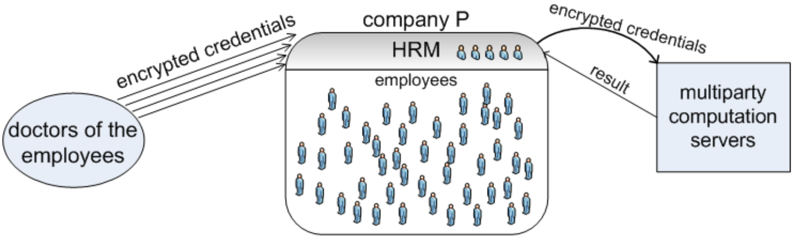
- HRM (human resource management) needs to mine privacy sensitive medical data from employees

Medical Data of Employees



- HRM (human resource management) needs to mine privacy sensitive medical data from employees

Medical Data of Employees



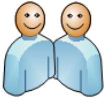
- HRM (human resource management) needs to mine privacy sensitive medical data from employees
- HRM can send these encrypted credentials to SFE servers for analysis

- ... encrypted DNA, encrypted parts of EPD (electronic patient dossier)

Boy or Girl?

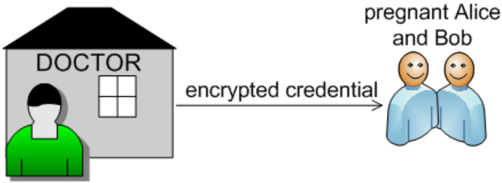
- Alice and Bob want to buy/receive clothes and toys for unborn baby

Boy or Girl?



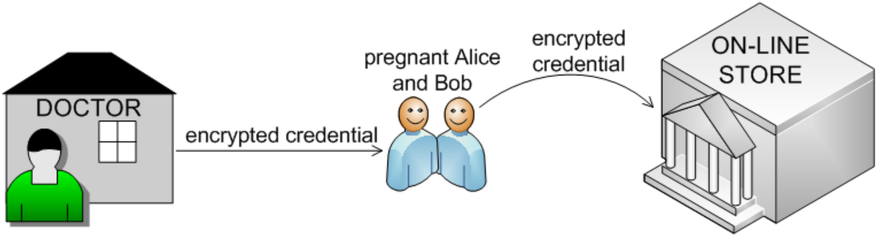
- Alice and Bob want to buy/receive clothes and toys for unborn baby
- They do not want to know the gender yet

Boy or Girl?



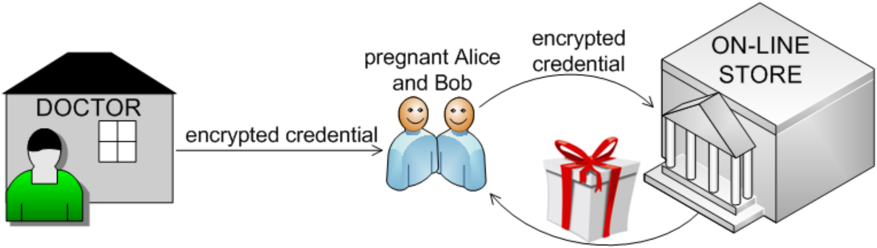
- Alice and Bob want to buy/receive clothes and toys for unborn baby
- They do not want to know the gender yet

Boy or Girl?



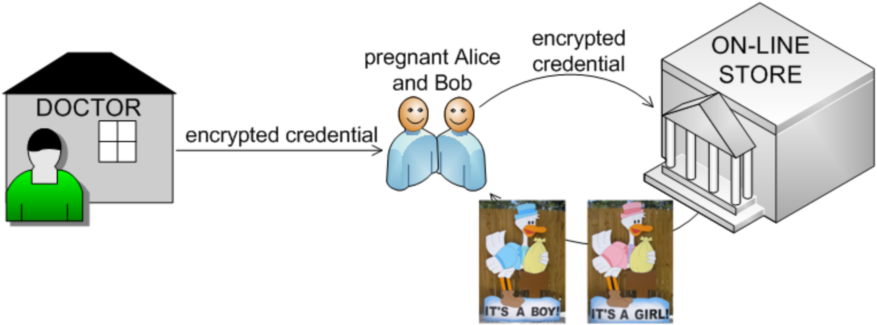
- Alice and Bob want to buy/receive clothes and toys for unborn baby
- They do not want to know the gender yet

Boy or Girl?



- Alice and Bob want to buy/receive clothes and toys for unborn baby
- They do not want to know the gender yet
- They unwrap the presents as soon as the baby is born!

Boy or Girl?



- Alice and Bob want to buy/receive clothes and toys for unborn baby
- They do not want to know the gender yet
- They unwrap the presents as soon as the baby is born!

- ... get yard signs in advance

Outline

ElGamal Cryptosystem

- We use the ElGamal cryptosystem:
  - Group $\langle g \rangle$ of prime order q
  - Secret key $\lambda \in_R \mathbb{Z}_q$, public key $f = g^\lambda$
  - Encryption of $x \in \mathbb{Z}_q$: $[\![x]\!] = (g^r, g^x f^r)$ for $r \in_R \mathbb{Z}_q$

ElGamal Cryptosystem

- We use the ElGamal cryptosystem:
  - Group $\langle g \rangle$ of prime order q
  - Secret key $\lambda \in_R \mathbb{Z}_q$, public key $f = g^\lambda$
  - Encryption of $x \in \mathbb{Z}_q$: $[\![x]\!] = (g^r, g^x f^r)$ for $r \in_R \mathbb{Z}_q$

- Homomorphic properties:
  - Addition: $[\![x]\!][\![y]\!] = [\![x+y]\!]$
  - Multiplication by constant: $[\![x]\!]^c = [\![xc]\!]$
  - Re-randomization: $[\![x]\!][\![0]\!] = [\![x]\!]$

Outline

Anonymous Credential Schemes

- Three components:
  - Key generation: algorithm for $\mathcal{I}$
  - Issuance: protocol for $(\mathcal{I}, \mathcal{U})$
  - Verification: protocol for $(\mathcal{U}, \mathcal{V})$

- Credentials are tuples (p, s, σ), where:
  - p public part, and σ signature on p
  - s secret part corresponding to p

- s contains the attributes on which the credential is issued
- In this presentation: 2 attributes (x_1, x_2)

Brands' Anonymous Credential Scheme

- Brands' credential schemes: single-use credentials
- We use the Brands' credential scheme based on the blind Chaum-Pedersen (CP) signature scheme
  - Issuance possible without $\mathcal{I}$ learning attributes
  - This variant is also used in U-Prove

Brands' Anonymous Credential Scheme

- Brands' credential schemes: single-use credentials
- We use the Brands' credential scheme based on the blind Chaum-Pedersen (CP) signature scheme
  - Issuance possible without $\mathcal{I}$ learning attributes
  - This variant is also used in U-Prove

- Key generation: group $\langle g \rangle$ of prime order q
  Secret key $x_0, y_1, y_2 \in_R \mathbb{Z}_q$, public $h_0 = g^{x_0}$, $g_1 = g^{y_1}$, $g_2 = g^{y_2}$

Brands' Anonymous Credential Scheme

- Brands' credential schemes: single-use credentials
- We use the Brands' credential scheme based on the blind Chaum-Pedersen (CP) signature scheme
  - Issuance possible without $\mathcal{I}$ learning attributes
  - This variant is also used in U-Prove

- Key generation: group $\langle g \rangle$ of prime order q
  Secret key $x_0, y_1, y_2 \in_R \mathbb{Z}_q$, public $h_0 = g^{x_0}$, $g_1 = g^{y_1}$, $g_2 = g^{y_2}$

- Credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^\alpha = h'$

Brands' Anonymous Credential Scheme

- Credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^{\alpha} = h'$

Brands' Anonymous Credential Scheme

- Credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  - a) σ is CP-signature on h'
  - b) $(g_1^{x_1} g_2^{x_2} h_0)^{\alpha} = h'$

- Issuance: $\mathcal{I}$ issues blind CP-signature σ on $h = (h')^{1/\alpha}$

$\mathcal{U}$		$\mathcal{I}$
(knows: x_1, x_2, α; h, h')		(knows: x_0; h)
		$\ldots$
	$\xleftarrow{z = h^{x_0}\,;\, a, b}$	
$\ldots$		
	$\xrightarrow{c}$	
		$\ldots$
	$\xleftarrow{r}$	
$\ldots$		

Brands' Anonymous Credential Scheme

- Credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^\alpha = h'$

- Issuance: $\mathcal{I}$ issues blind CP-signature σ on $h = (h')^{1/\alpha}$

$$
\begin{array}{ccc}
\mathcal{U} & & \mathcal{I} \\
(\text{knows: } x_1, x_2, \alpha; h, h') & & (\text{knows: } x_0; h) \\
 & & \ldots \\
 & \xleftarrow{z = h^{x_0};\ a, b} & \\
\ldots & & \\
 & \xrightarrow{c} & \\
 & & \ldots \\
 & \xleftarrow{r} & \\
\ldots & &
\end{array}
$$

- Verification: $\mathcal{U}$ proves knowledge of x_1, x_2, α s.t. $(g_1^{x_1} g_2^{x_2} h_0)^\alpha = h'$

Extension to Encrypted Attributes

- We extend Brands' credential scheme with encrypted attributes

- Several variations discussed in the paper, where
  - $\rightarrow$ none of the participants learns the attributes
  - $\rightarrow$ all parties learn a specific (possibly different) set of attributes
  - $\rightarrow$ $\mathcal{I}$ learns the attributes, but $\mathcal{U}, \mathcal{V}$ *do not* learn these

- Now: simplified version of the encrypted credential scheme

Extension to Encrypted Attributes

- Brands' credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^{\alpha} = h'$

Extension to Encrypted Attributes

- Brands' credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^{\alpha} = h'$

- Encrypted credential on (c_1, c_2) is a tuple $(\underbrace{h', c_1', c_2'}_{p}, \underbrace{\alpha}_{s}, \sigma)$ s.t.:
  a)
  b)

- Now, encryptions $c_1 = [\![x_1]\!]$, $c_2 = [\![x_2]\!]$ belong to public part

Extension to Encrypted Attributes

- Brands' credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^\alpha = h'$

- Encrypted credential on (c_1, c_2) is a tuple $(\underbrace{h', c_1', c_2'}_{p}, \underbrace{\alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on (h', c_1', c_2')
  b)

- Now, encryptions $c_1 = [\![x_1]\!]$, $c_2 = [\![x_2]\!]$ belong to public part
  $\rightarrow \mathcal{U}$ has to re-randomize c_1, c_2 in issuance

Extension to Encrypted Attributes

- Brands' credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on h'
  b) $(g_1^{x_1} g_2^{x_2} h_0)^\alpha = h'$

- Encrypted credential on (c_1, c_2) is a tuple $(\underbrace{h', c_1', c_2'}_{p}, \underbrace{\alpha}_{s}, \sigma)$ s.t.:
  a) σ is CP-signature on (h', c_1', c_2')
  b) $(D((c_1')^{y_1}(c_2')^{y_2})h_0)^\alpha = h'$

- Now, encryptions $c_1 = [\![x_1]\!]$, $c_2 = [\![x_2]\!]$ belong to public part
  $\rightarrow \mathcal{U}$ has to re-randomize c_1, c_2 in issuance
  $\rightarrow \mathcal{U}$ cannot prove knowledge of x_1, x_2 in verification

Technical Issues

$\mathcal{U}$ has to re-randomize $c_1 = [\![x_1]\!], c_2 = [\![x_2]\!]$ in issuance

$\mathcal{U}$ cannot prove knowledge of x_1, x_2 in verification

Technical Issues

$\mathcal{U}$ has to re-randomize $c_1 = [\![x_1]\!], c_2 = [\![x_2]\!]$ in issuance

Problem: user can replace c_1 by $[\![0]\!]$ and obtain oracle for "$x_1 = 0$?"

$\mathcal{U}$ cannot prove knowledge of x_1, x_2 in verification

Technical Issues

$\mathcal{U}$ has to re-randomize $c_1 = [\![x_1]\!], c_2 = [\![x_2]\!]$ in issuance

Problem: user can replace c_1 by $[\![0]\!]$ and obtain oracle for "$x_1 = 0$?"
Solution: credential will actually be issued on $[\![x_i + \phi_i]\!]$, for $\phi_i \in_R \mathbb{Z}_q$

- g^{ϕ_i} can be made public
- $[\![x_i]\!]$ can be obtained from $[\![x_i + \phi_i]\!]$ and g^{ϕ_i}

$\mathcal{U}$ cannot prove knowledge of x_1, x_2 in verification

Technical Issues

$\mathcal{U}$ has to re-randomize $c_1 = [\![x_1]\!], c_2 = [\![x_2]\!]$ in issuance

Problem: user can replace c_1 by $[\![0]\!]$ and obtain oracle for "$x_1 = 0$?"
Solution: credential will actually be issued on $[\![x_i + \phi_i]\!]$, for $\phi_i \in_R \mathbb{Z}_q$

- g^{ϕ_i} can be made public
- $[\![x_i]\!]$ can be obtained from $[\![x_i + \phi_i]\!]$ and g^{ϕ_i}

$\mathcal{U}$ cannot prove knowledge of x_1, x_2 in verification

Verification: $\mathcal{U}$ proves knowledge of α s.t. $(D((c_1')^{y_1}(c_2')^{y_2})h_0)^{\alpha} = h'$

- Verifier needs secret data, namely y_1, y_2, and λ (for decryption)
- Verifier is required to be semi-honest (threshold cryptography)

Security Analysis

- Proven secure, against:
  - Malicious $\mathcal{I}$ and $\mathcal{U}$
  - Semi-honest $\mathcal{V}$ (threshold cryptography)

- Security based on:
  - DDH assumption
  - Random Oracle Model
  - Security of blind Chaum-Pedersen scheme
  - A new assumption
- Same level of security as original Brands' scheme

Conclusions

- We introduced anonymous credential schemes with encrypted attributes, and presented and analyzed various efficient constructions based on a credential scheme by Brands
- Wide range of applications:
  - Missing link between credential schemes and SFE
  - Medical data of employees, boy or girl?, ...
  - Letters of recommendation, medical data of illnesses, ...
- Further research:
  - Encrypted credential schemes with multi-use credentials
  - Public verifiability of encrypted credentials

Thank you for your attention!

SUPPORTING SLIDES!!!

Σ-protocols

- Simplest example: Schnorr's identification protocol
- We consider a group $\langle g \rangle$, and public $h \in \langle g \rangle$
- Prover wants to prove that he knows $x = \log_g h$

Σ-protocols

- Simplest example: Schnorr's identification protocol
- We consider a group $\langle g \rangle$, and public $h \in \langle g \rangle$
- Prover wants to prove that he knows $x = \log_g h$

$$
\begin{array}{rcl}
\textbf{Prover} & & \textbf{Verifier} \\
\text{(knows: } h; x\text{)} & & \text{(knows: } h\text{)} \\
u \in_R \mathbb{Z}_q, \quad a \leftarrow g^u & \xrightarrow{\quad a \quad} & \\
 & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_q \\
r \leftarrow u + cx \bmod q & \xrightarrow{\quad r \quad} & \\
 & & g^r \stackrel{?}{=} ah^c
\end{array}
$$

Σ-protocols

- Simplest example: Schnorr's identification protocol
- We consider a group $\langle g \rangle$, and public $h \in \langle g \rangle$
- Prover wants to prove that he knows $x = \log_g h$

Prover		**Verifier**
(knows: $h; x$)		(knows: h)
$u \in_R \mathbb{Z}_q, \quad a \leftarrow g^u$	$\xrightarrow{a}$	
	$\xleftarrow{c}$	$c \in_R \mathbb{Z}_q$
$r \leftarrow u + cx \bmod q$	$\xrightarrow{r}$	
		$g^r \stackrel{?}{=} ah^c$

- This is Σ-protocol for relation $\{(h; x) \mid h = g^x\}$
- Σ-protocol: completeness, special-soundness, honest-verifier zero-knowledge

Key Generation for $\mathcal{I}$

- **Public:** group $\langle g \rangle$ of prime order q; $h_0 \in \langle g \rangle$, $g_1, g_2 \in_R \langle g \rangle$
- **Secret:** $x_0 \in_R \mathbb{Z}_q$ such that $h_0 = g^{x_0}$

- A credential on (x_1, x_2) is a tuple $(\underbrace{h'}_{p}, \underbrace{x_1, x_2, \alpha}_{s}, \underbrace{z', c', r'}_{\sigma})$ s.t.:

$$c' = \mathcal{H}(h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'}) \text{ and } (g_1^{x_1} g_2^{x_2} h_0)^{\alpha} = h'$$

Issuance for $(\mathcal{I}, \mathcal{U})$

- For (x_1, x_2), $\mathcal{U}$ and $\mathcal{I}$ compute $h = g_1^{x_1} g_2^{x_2} h_0$

$$
\begin{array}{rcl}
\mathcal{U} & & \mathcal{I} \\
 & & z \leftarrow h^{x_0}, \quad w \in_R \mathbb{Z}_q \\
 & \xleftarrow{\quad z;a,b \quad} & a \leftarrow g^w, \quad b \leftarrow h^w \\
\alpha \in_R \mathbb{Z}_q^*, \quad \beta, \gamma \in_R \mathbb{Z}_q & & \\
h' \leftarrow h^\alpha, \quad z' \leftarrow z^\alpha & & \\
a' \leftarrow h_0^\beta g^\gamma a & & \\
b' \leftarrow (z')^\beta (h')^\gamma b^\alpha & & \\
c' \leftarrow \mathcal{H}(h', z', a', b') & & \\
c \leftarrow c' + \beta \bmod q & \xrightarrow{\quad c \quad} & \\
 & \xleftarrow{\quad r \quad} & r \leftarrow cx_0 + w \bmod q \\
a \stackrel{?}{=} g^r h_0^{-c}, \quad b \stackrel{?}{=} h^r z^{-c} & & \\
r' \leftarrow r + \gamma \bmod q & &
\end{array}
$$

- Note: $c' = \mathcal{H}(h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'})$ and $(g_1^{x_1} g_2^{x_2} h_0)^\alpha = h'$

Verification for $(\mathcal{U}, \mathcal{V})$

- Brands' credential is $(h', x_1, x_2, \alpha, z', c', r')$ such that:

$$c' = \mathcal{H}(h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'}) \text{ and } (g_1^{x_1} g_2^{x_2} h_0)^{\alpha} = h'$$

$$\begin{array}{rcl}
\mathcal{U} & & \mathcal{V} \\
\text{(knows: } h', z', c', r'; x_1, x_2, \alpha) & & \\
u_1, u_2, u_\alpha \in_R \mathbb{Z}_q & & \\
a \leftarrow (h')^{u_\alpha} g_1^{-u_1} \cdots g_l^{-u_l} & \xrightarrow{a; h', z', c', r'} & \\
 & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_q \\
(r_i \leftarrow u_i + c x_i \bmod q)_{i=1}^2 & & \\
r_\alpha \leftarrow u_\alpha + c\alpha^{-1} \bmod q & \xrightarrow{r_1, r_2, r_\alpha} & \\
 & & c' \stackrel{?}{=} \mathcal{H}(h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'}) \\
 & & (h')^{r_\alpha} g_1^{-r_1} \cdots g_l^{-r_l} \stackrel{?}{=} a h_0^c
\end{array}$$

- Σ-protocol for $\{(h'; x_1, x_2, \alpha) \mid h_0 = (h')^{\alpha^{-1}} g_1^{-x_1} g_2^{-x_2} \wedge \alpha \neq 0\}$

Key Generation for $(\mathcal{I}, \mathcal{V})$

- **Public:** group $\langle g \rangle$ of prime order q; $h_0, g_1, g_2, f, \hat{f}, f_1 \in \langle g \rangle$
- **Secret:** $x_0, \phi_1 \in_R \mathbb{Z}_q$ for $\mathcal{I}$ and $y_1, y_2, \lambda \in_R \mathbb{Z}_q$ for $\mathcal{V}$ such that

$$h_0 = g^{x_0} \quad g_1 = g^{y_1} \quad g_2 = g^{y_2} \quad f = g^{\lambda} \quad \hat{f} = f^{x_0} = h_0^{\lambda} \quad f_1 = g^{\phi_1}$$

- A credential on x_1^* is a tuple $(\underbrace{h', c_1', c_2'}_{p}, \underbrace{\alpha}_{s}, \underbrace{z', z_1', z_2', c', r'}_{\sigma})$, where $c_1' = [\![x_1^* + \phi_1]\!]$, such that:

$$c' = \mathcal{H}([c_i', z_i', (c_i')^{r'}(z_i')^{-c'}]_{i=1}^{2}; h', z', g^{r'}h_0^{-c'}, (h')^{r'}(z')^{-c'})$$

$$\text{and } (D((c_1')^{y_1}(c_2')^{y_2})h_0)^{\alpha} = h'$$

- Note the transformation from $[\![x_1^*]\!]$ to $[\![x_1^* + \phi_1]\!]$
  - Otherwise, a malicious $\mathcal{U}'$ can replace c_1' by $[\![0]\!]$
  - Verification succeeds *if and only if* $x_1^* = 0$
    $\rightarrow$ this gives $\mathcal{U}'$ an oracle for '$x_1^* = 0$?'

Issuance for $(\mathcal{I}, \mathcal{U})$

- Credential issuance on $x_1^* \in \{0,1\}$

$$
\begin{array}{lcl}
\mathcal{U} & & \mathcal{I} \\
 & & r_1, r_2, x_I \in_R \mathbb{Z}_q \\
 & & c_1 \leftarrow (g^{r_1}, g^{x_1^*} f_1 f^{r_1}) \\
 & & c_2 \leftarrow (g^{r_2}, g^{x_2} f^{r_2}) \\
 & & h \leftarrow g_1^{x_1^*+\phi_1} g_2^{x_2} h_0 \\
 & & z \leftarrow h^{x_0}, \quad (z_i \leftarrow c_i^{x_0})_{i=1}^2 \\
 & & w \in_R \mathbb{Z}_q, \quad a \leftarrow g^w, \quad b \leftarrow h^w \\
 & & \tilde{f} \leftarrow f^w, \quad (e_i \leftarrow c_i^w)_{i=1}^2 \\
 & \xleftarrow{\substack{h, z, (c_i, z_i)_{i=1}^2; \\ a, b, \tilde{f}, (e_i)_{i=1}^2}} & \\
\ldots & &
\end{array}
$$

- Notice that also $z_i = c_1^{x_0}$ and $e_{i0} = c_i^{w_0}$ are computed

$\rightarrow$

Issuance for $(\mathcal{I}, \mathcal{U})$

- $\mathcal{U}$ and $\mathcal{I}$ know $h, z, c_1, c_2, z_1, z_2, a_0, b_0, \tilde{f}, (e_{i0})_{i=1}^2$

$$\alpha \in_R \mathbb{Z}_q^*, \quad \beta, \gamma \in_R \mathbb{Z}_q$$

$$h' \leftarrow h^\alpha, \quad z' \leftarrow z^\alpha$$

$$a' \leftarrow h_0^\beta g^\gamma a, \quad b' \leftarrow (z')^\beta (h')^\gamma b^\alpha$$

$$\left(\begin{array}{r} \delta_i \in_R \mathbb{Z}_q, \quad c_i' \leftarrow c_i \cdot (g, f)^{\delta_i} \\ z_i' \leftarrow z_i \cdot (h_0, \hat{f})^{\delta_i} \\ e_i' \leftarrow (z_i')^\beta (c_i')^\gamma e_i \cdot (a, \tilde{f})^{\delta_i} \end{array} \right)_{i=1}^2$$

$$c' \leftarrow \mathcal{H}([c_i', z_i', e_i']_{i=1}^2; h', z', a', b')$$

$$c \leftarrow c' + \beta \bmod q \quad \xrightarrow{\ c\ }$$

$$\xleftarrow{\ r\ } \quad r \leftarrow cx_0 + w \bmod q$$

$$a \stackrel{?}{=} g^r h_0^{-c}, \quad b \stackrel{?}{=} h^r z^{-c}$$

$$\tilde{f} \stackrel{?}{=} f^r \hat{f}^{-c}, \quad (e_i \stackrel{?}{=} c_i^r z_i^{-c})_{i=1}^2$$

$$r' \leftarrow r + \gamma \bmod q$$

Verification for $(\mathcal{U}, \mathcal{V})$

- $\mathcal{U}$ proves knowledge of α such that $(D((c_1')^{y_1}(c_2')^{y_2})h_0)^{\alpha} = h'$

$$
\begin{array}{rcl}
\mathcal{U} & & \mathcal{V} \\
\text{(knows: } h', c_1', c_2'; \alpha) & & \text{(knows: } h', c_1', c_2'; y_1, y_2, \lambda) \\
u \in_R \mathbb{Z}_q, \quad a \leftarrow (h')^u & \xrightarrow{a} & \\
 & \xleftarrow{c} & c \in_R \mathbb{Z}_q \\
r \leftarrow u + c\alpha^{-1} \bmod q & \xrightarrow{r} & \\
 & & b \leftarrow \left[(h')^r \stackrel{?}{=} a(D((c_1')^{y_1}(c_2')^{y_2})h_0)^c\right] \\
 & \xleftarrow{b} &
\end{array}
$$

- Protocol is a zero-knowledge proof of knowledge for α such that $(D((c_1')^{y_1}(c_2')^{y_2})h_0)^{\alpha} = h'$
- $\mathcal{V}$ is required to be semi-honest (threshold cryptography)

Verification for $(\mathcal{U}, \mathcal{V})$

- $\mathcal{U}$ proves knowledge of α such that $(D((c_1')^{y_1}(c_2')^{y_2})h_0)^{\alpha} = h'$

$$
\begin{array}{ccc}
\mathcal{U} & & \mathcal{V} \\
(\text{knows: } h', c_1', c_2'; \alpha) & & (\text{knows: } h', c_1', c_2'; y_1, y_2, \lambda) \\
u \in_R \mathbb{Z}_q, \quad a \leftarrow (h')^u & \xrightarrow{\quad a \quad} & \\
 & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_q \\
r \leftarrow u + c\alpha^{-1} \bmod q & \xrightarrow{\quad r \quad} & \\
 & \xleftarrow{\quad b \quad} & b \leftarrow \left[(h')^r \stackrel{?}{=} a(D((c_1')^{y_1}(c_2')^{y_2})h_0)^c\right]
\end{array}
$$

- Protocol is a zero-knowledge proof of knowledge for α such that $(D((c_1')^{y_1}(c_2')^{y_2})h_0)^{\alpha} = h'$
- $\mathcal{V}$ is required to be semi-honest (threshold cryptography)
- $\mathcal{V}$ can obtain $[\![x_1^*]\!]$ by computing $c_1' \cdot (1, f_1^{-1})$:

$$c_1' \cdot (1, f_1^{-1}) = (g^r, g^{x_1^* + \phi_1} f^r) \cdot (1, g^{-\phi_1}) = (g^r, g^{x_1^*} f^r) = [\![x_1^*]\!]$$

Assumption

- $\mathcal{U}'$ is issued K credentials on x_j^* $(j = 1, \ldots, K)$, and learned $(c_{ji})_{i=1}^2$
- Then he outputs a tuple $(h', c_1', c_2, \alpha, z', z_1', z_2', c', r')$. Now, either
  - This tuple is not a valid credential
  - There exists a j such that

$\mathcal{U}'$ knows values β_1, β_2 such that $(c_i')_{i=1}^2 = (c_{ji}(g, f)^{\beta_i})_{i=1}^2$

One-more Unforgeability

- ‘Hard to obtain $K+1$ credentials after K issuing executions’
- Credential scheme is based on blind Chaum-Pedersen signature scheme
- Reducing one-more forgeries:

$$\begin{array}{ccccc} \mathcal{U}' & & \mathcal{F} & & \mathcal{S}_{CP} \\ \left[\right. & \xleftarrow{\text{issues credential}} & & \xleftarrow{\text{issues CP-signature}} & \left.\right]\ K \text{ times} \\ & \xrightarrow{\text{forgery: } K+1 \text{ credentials}} & & \xrightarrow{\text{forgery: } K+1 \text{ CP-signatures}} & \end{array}$$

- Our scheme is at least as secure against one-more forgeries

Verification Protocol

- Recall the verification protocol

$$
\begin{array}{rcl}
\mathcal{U} & & \mathcal{V} \\
(\text{knows: } h', c_1', c_2'; \alpha) & & (\text{knows: } h', c_1', c_2'; y_1, y_2, \lambda) \\
u \in_R \mathbb{Z}_q, \quad a \leftarrow (h')^u & \xrightarrow{\quad a \quad} & \\
 & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_q \\
r \leftarrow u + c\alpha^{-1} \bmod q & \xrightarrow{\quad r \quad} & \\
 & \xleftarrow{\quad b \quad} & b \leftarrow \left[(h')^r \stackrel{?}{=} a(D((c_1')^{y_1}(c_2')^{y_2})h_0)^c \right]
\end{array}
$$

- Protocol should be a secure proof of knowledge
  - Proof of knowledge: *complete* and *special sound*
  - Secure: views on protocol *simulateable* for passive $\mathcal{V}'$ and active $\mathcal{U}'$
- Simulation of view of active $\mathcal{U}'$: after sending r, $\mathcal{U}'$ already knows b