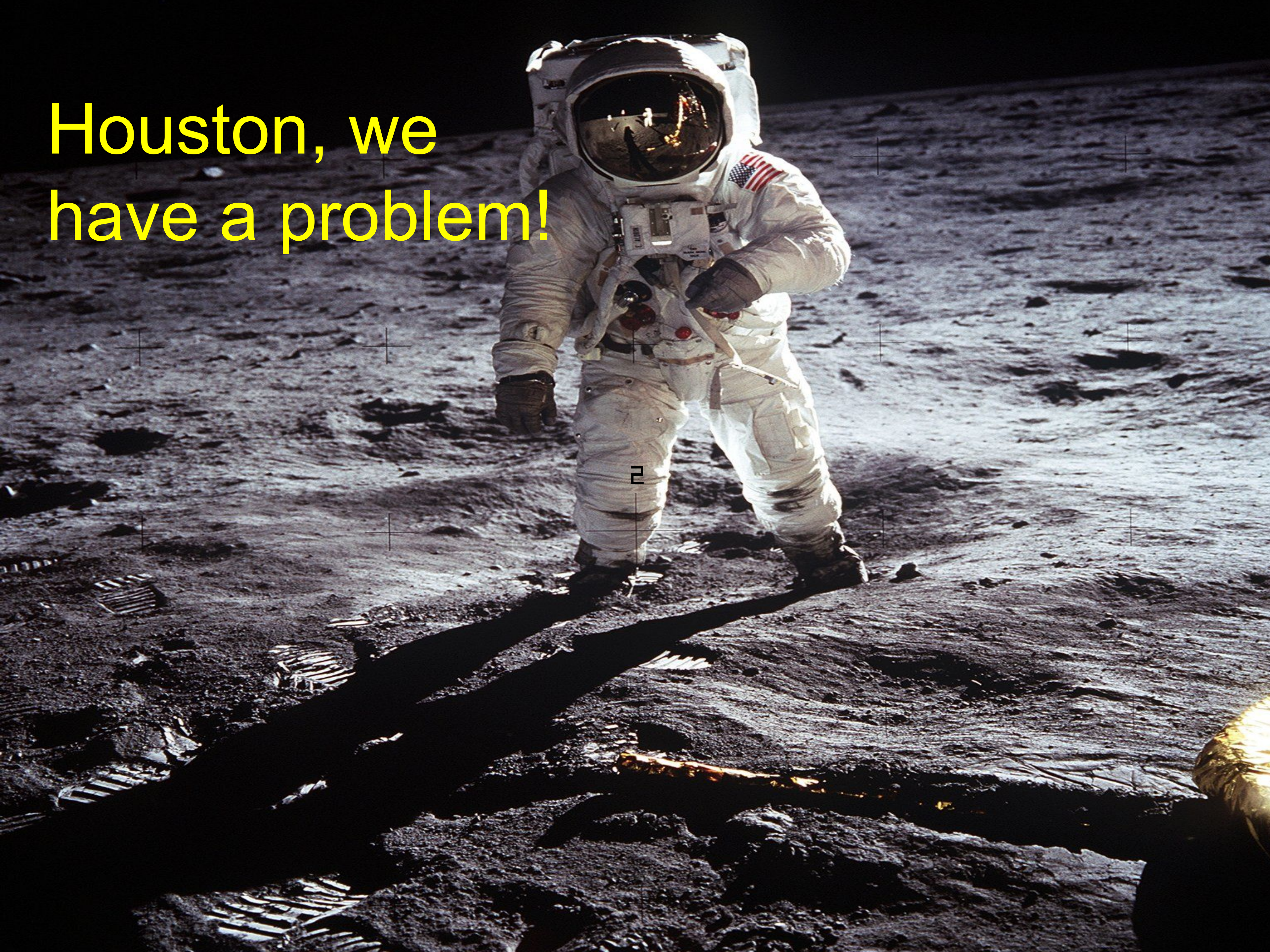
Anonymous Credentials

Jan Camenisch
IBM Research – Zurich

@jancamenisch
www.camenisch.org/eprivacy


Houston, we have a problem!


Houston, we
have a problem!
“Buzz Aldrin's footprints are still up there”
(Robin Wilton)

Computers don't forget

- Data storage ever cheaper → “store by default”
  - also collateral collection, surveillance cameras, Google Street View with wireless traffic, Apple location history,...
- Data mining ever better
  - self-training algorithms cleverer than their designers
  - not just trend detection, even prediction, e.g., flu pandemics, ad clicks, purchases,…
  - what about health insurance, criminal behavior?
- The world as we know it
  - Humans forget most things too quickly
  - Paper collects dust in drawers

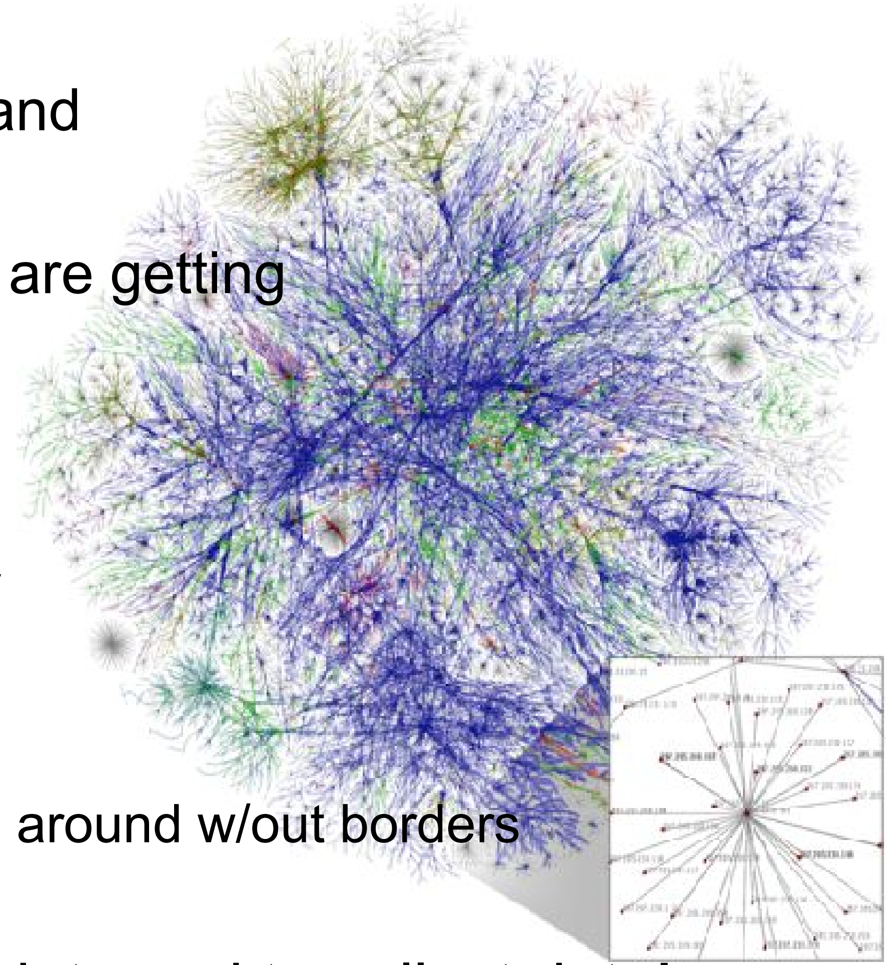
We build apps with the paper-based world in mind :-(

- *if it works it works*
- *security too often still an afterthought*
- *implementors too often have no crypto education*

Where's all my data?

The ways of data are hard to understand

- Devices, operating systems, & apps are getting more complex and intertwined
  - Mashups, Ad networks
  - Not visible to users, and experts
  - Data processing changes constantly
- And the cloud makes it worse...
  - Processing machines can be moved around w/out borders



Far too easy to lose (control over) data and to collect data!

You have no privacy, get over it .....?!?

… "The NSA has all our data anyway"

… "I have nothing to hide!"

- Huge security problem!
  - Millions of hacked passwords (100'000 followers $115 - 2013)
  - Stolen identities ($150 - 2005, $15 - 2009, $5 – 2013)

- Difficult to put figures down
  - Credit card fraud
  - Spam & marketing
  - Manipulating stock ratings, etc..
  - (Industrial) espionage

- We know secret services can do it easily, but they are not the only ones
  - but this is not about homeland security
  - and there are limits to the degree of protection that one can achieve

- last but not least: data are the new money, so they need to be protected!

No, but we need paradigm shift &

build stuff for the moon

rather than the sandy beach!

Need to protect our data!

- devices, sensors, etc cannot all be physically protected
  - authentication of all devices
  - authentication of all data

    ...makes it even worse :-(

- data cannot be controlled
  - minimize information
  - encrypt information
  - attach usage policies to each bit

So what can we do

- Legal approach
  - Regulate what information can be collected
  - How to collect it
  - How to use and protect it
  - Issue fines for misbehavior
  - Very different for different countries and cultures

- Technological approach
  - Protect data by encryption
  - Govern data by policies
  - Minimize data that needs to be used

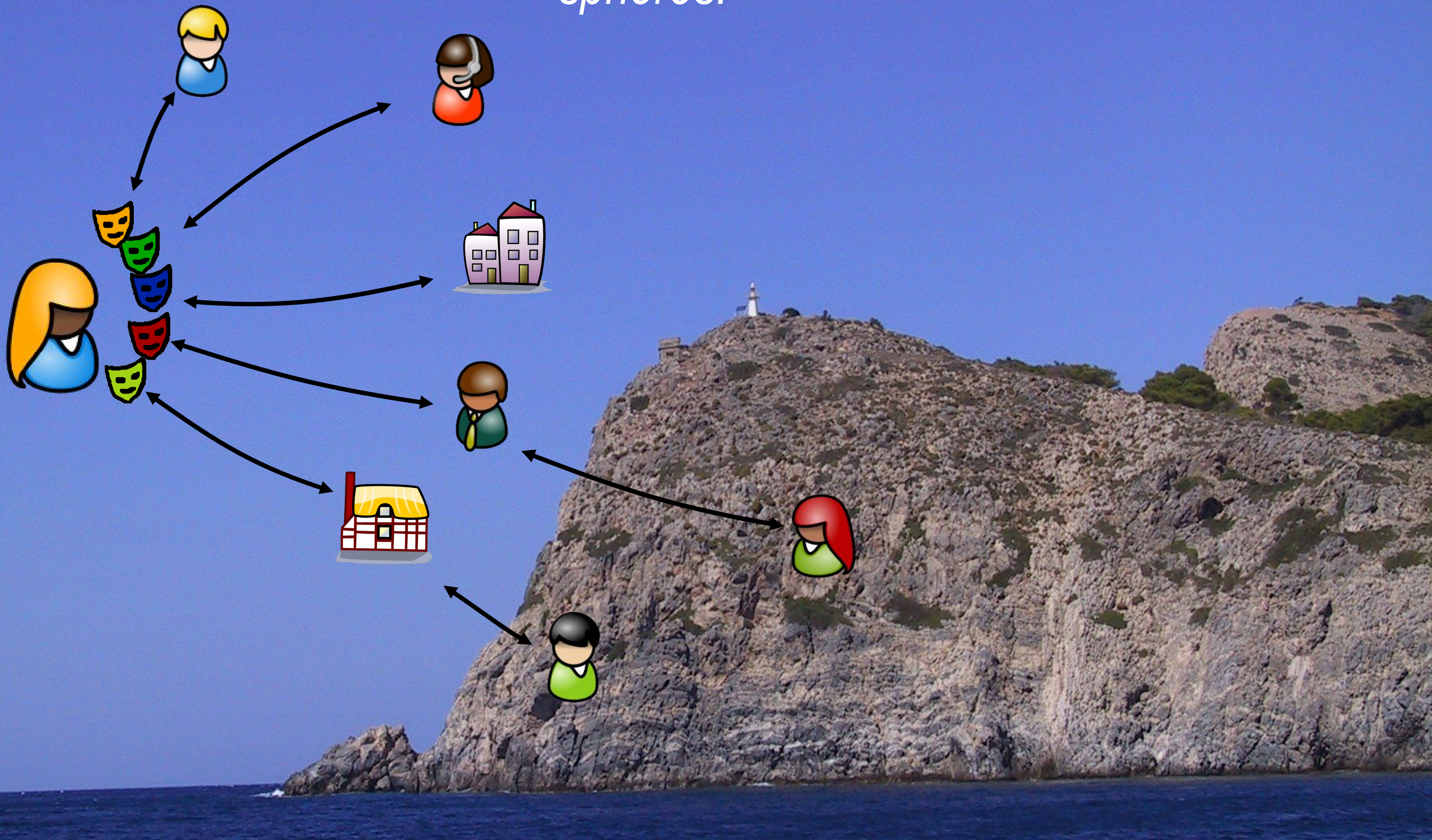
# of course there are limits...

- tracing is so easy
  - each piece of hardware is quite unique
  - log files everywhere
- …. but that's not the point!
  - it's not about NSA et al.
  - active vs. passive “adversaries”

..... still, *privacy by design!*

Our Vision

In the Information Society, users can act and interact in a safe and secure way while retaining control of their private spheres.



PETs Can Help!

Privacy, Identity, and Trust Mgmt Built-In Everywhere!

- Network Layer Anonymity
  - ... in mobile phone networks
  - ... in the Future Internet as currently discussed
  - ... access points for ID cards

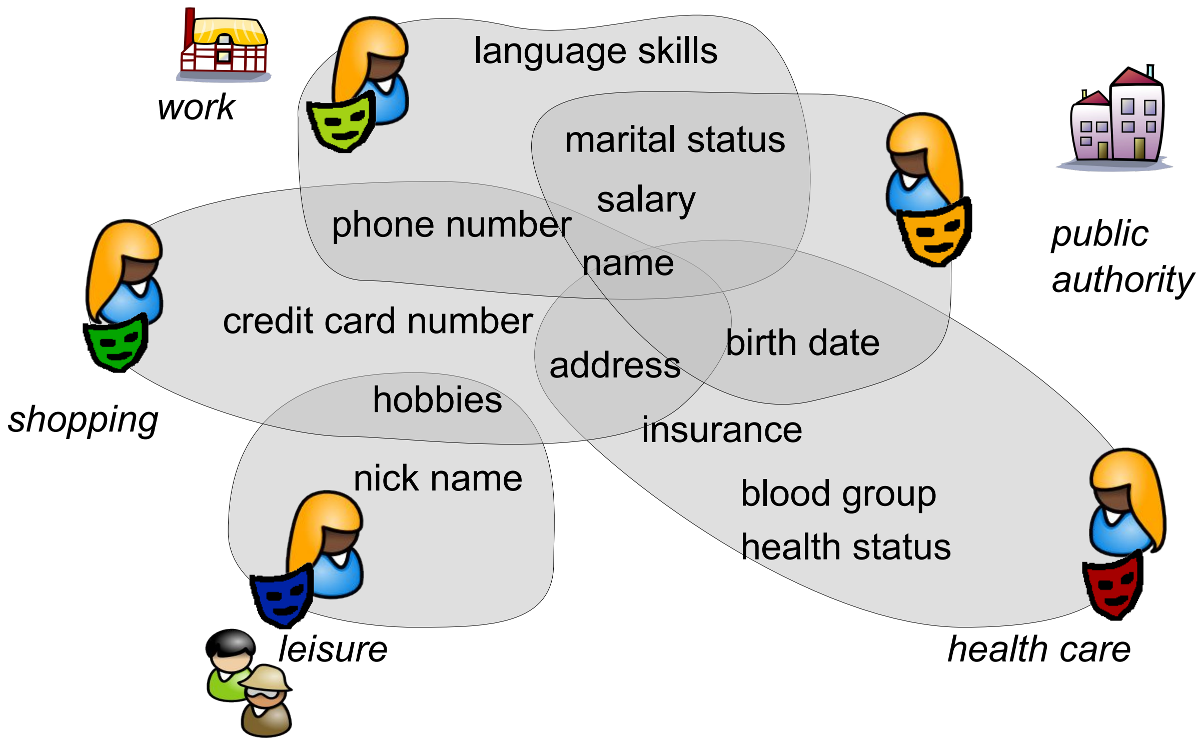
- Identification Layer
  - Access control & authorization

- Application Layer
  - "Standard" e-Commerce
  - Specific Apps, e.g., eVoting, OT, PIR, ...
  - Web 2.0, e.g., Facebook, Twitter, Wikis, ...

Privacy at the Authentication Layer

Authentication without identification

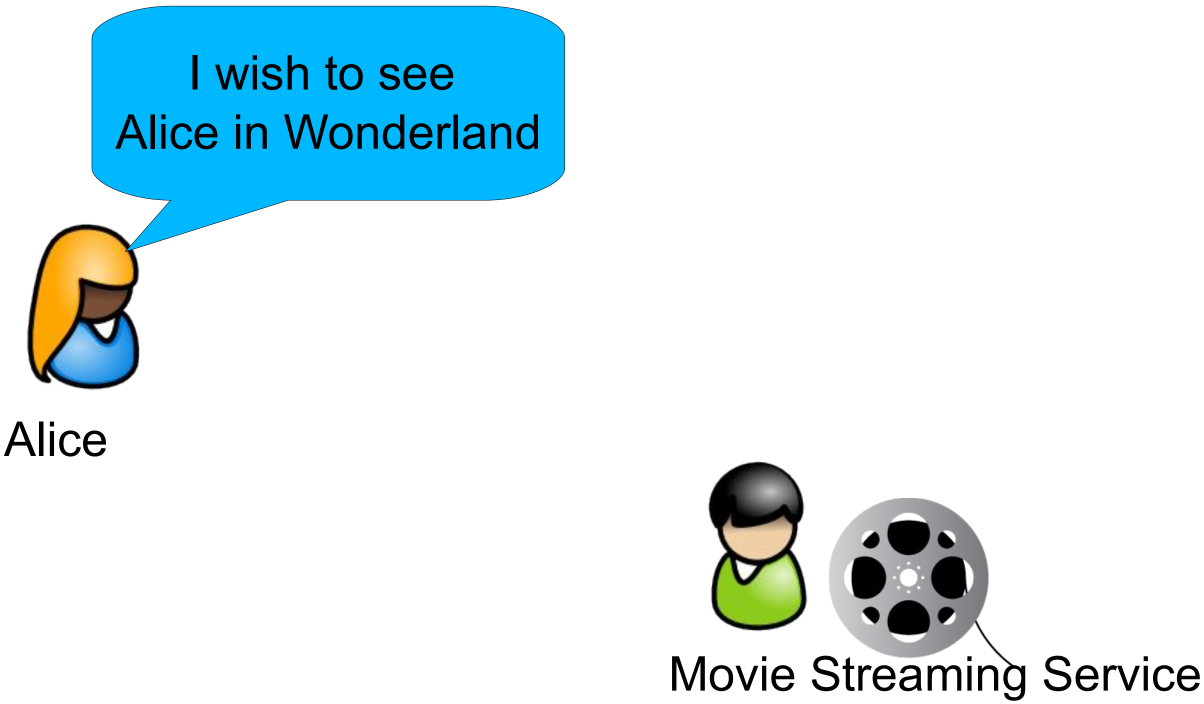
What is an identity & identity management?



- ID: set of attributes shared w/ someone
  - attributes are not static: user & party can add
- ID Management: two things to make ID useful
  - authentication means
  - means to transport attributes between parties

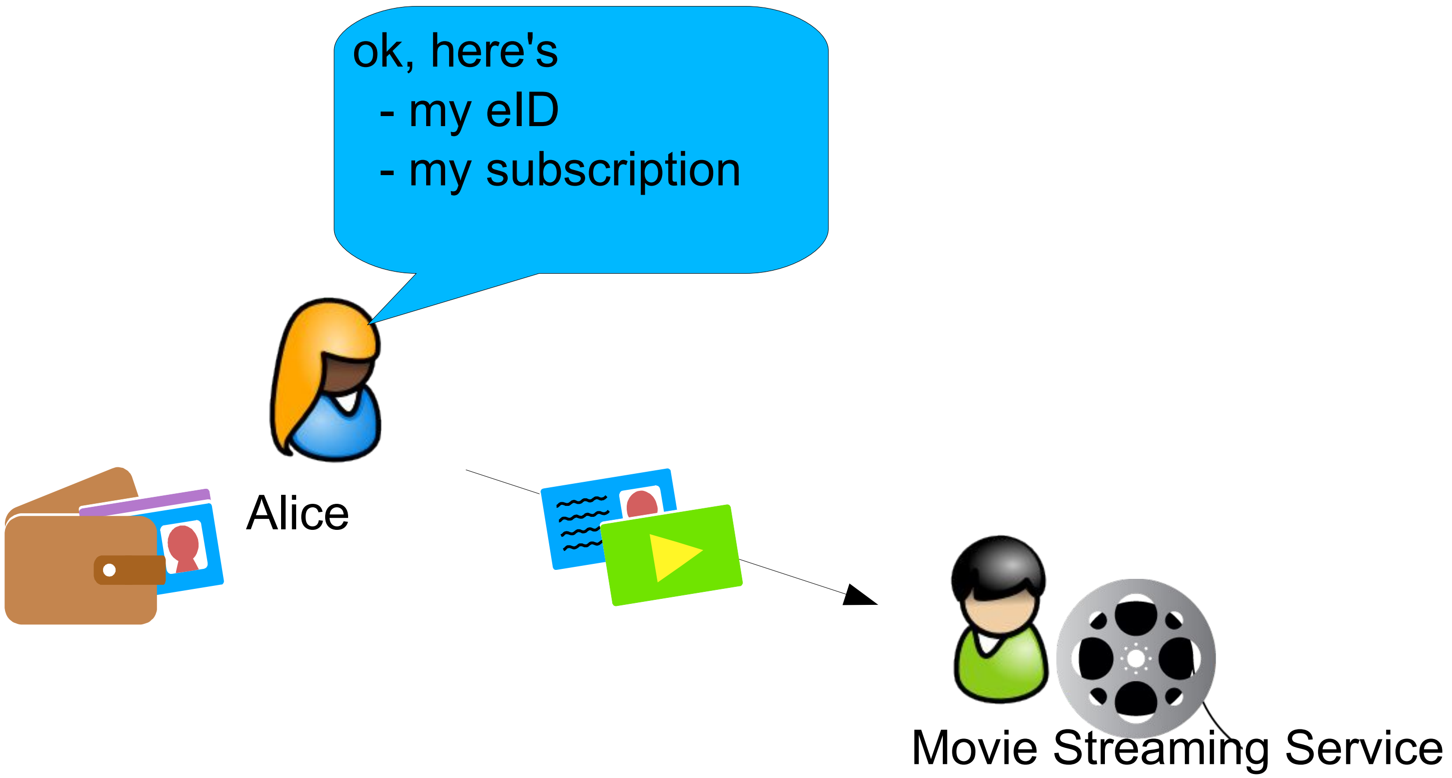
Let's see a scenario

Alice wants to watch a movie at Mplex

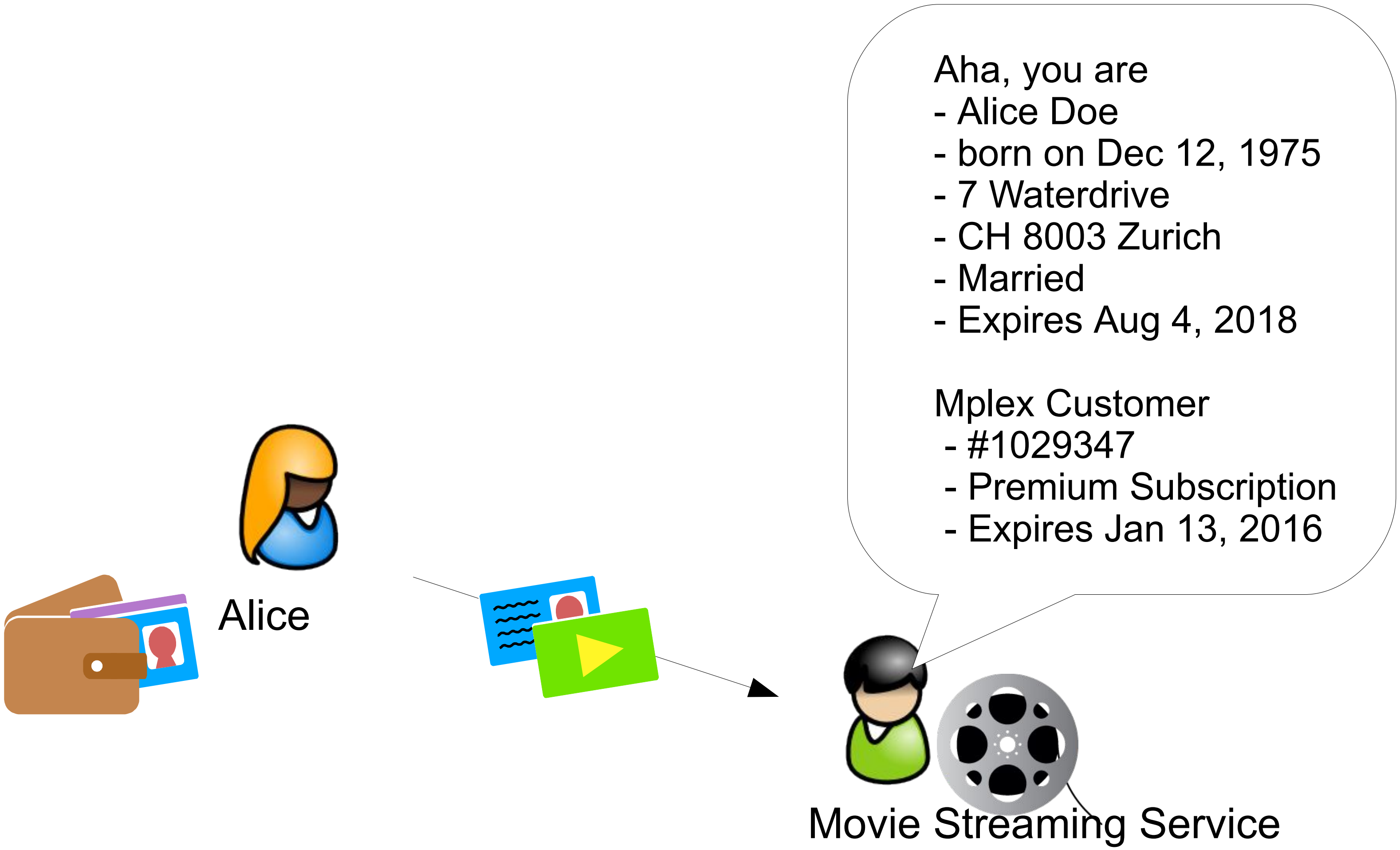


Alice wants to watch a movie at Mplex

Watching the movie with the traditional solution



Watching the movie with the traditional solution

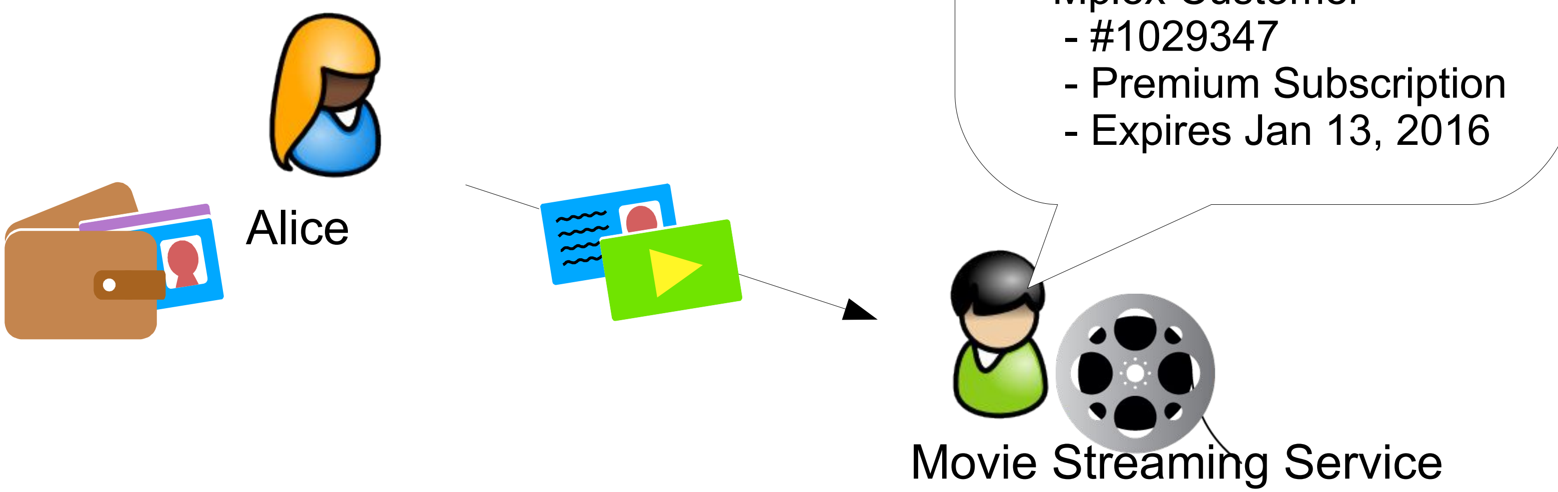


This is a privacy and security problem!

- - identity theft
- - profiling
- - discrimination

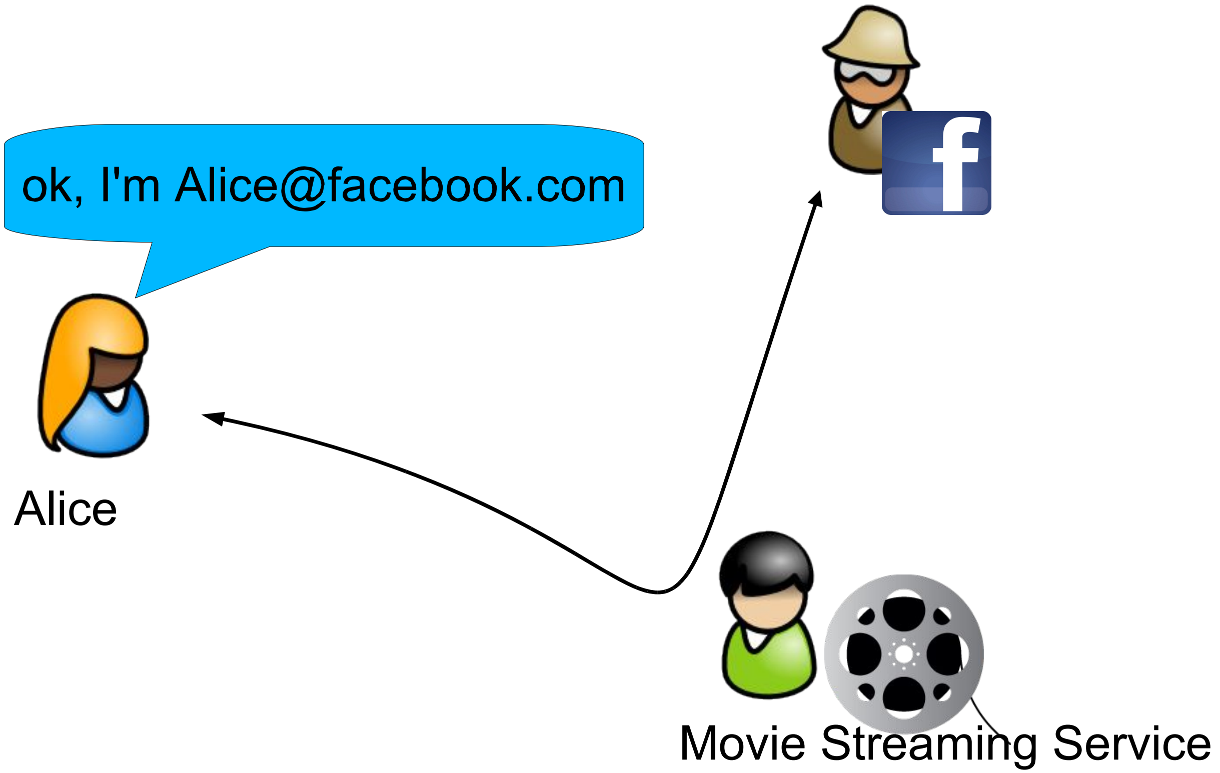
Aha, you are
- Alice Doe
- born on Dec 12, 1975
- 7 Waterdrive
- CH 8003 Zurich
- Married
- Expires Aug 4, 2018

Mplex Customer
- #1029347
- Premium Subscription
- Expires Jan 13, 2016



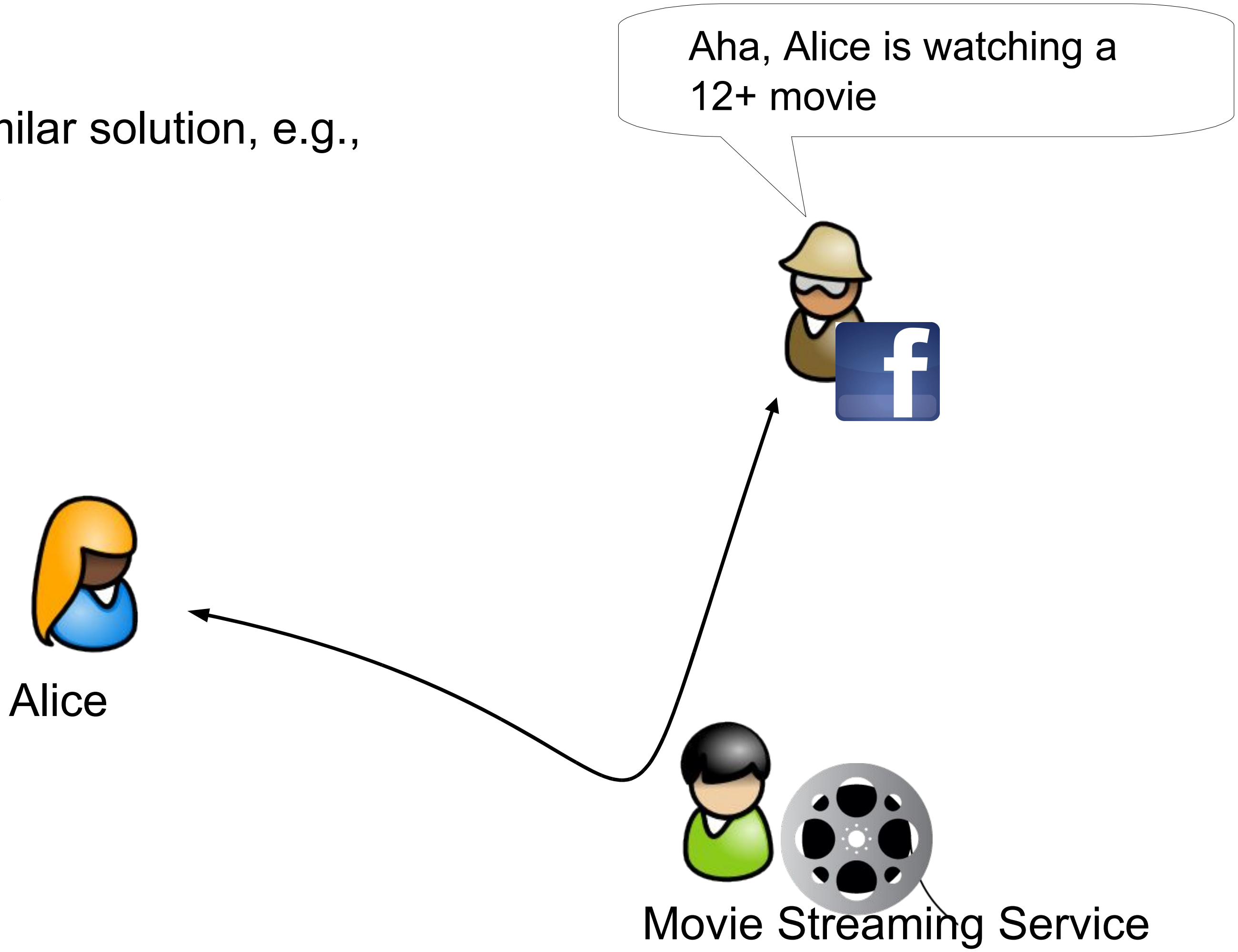
Watching the movie with the traditional solution

With OpenID and similar solution, e.g., log-in with Facebook



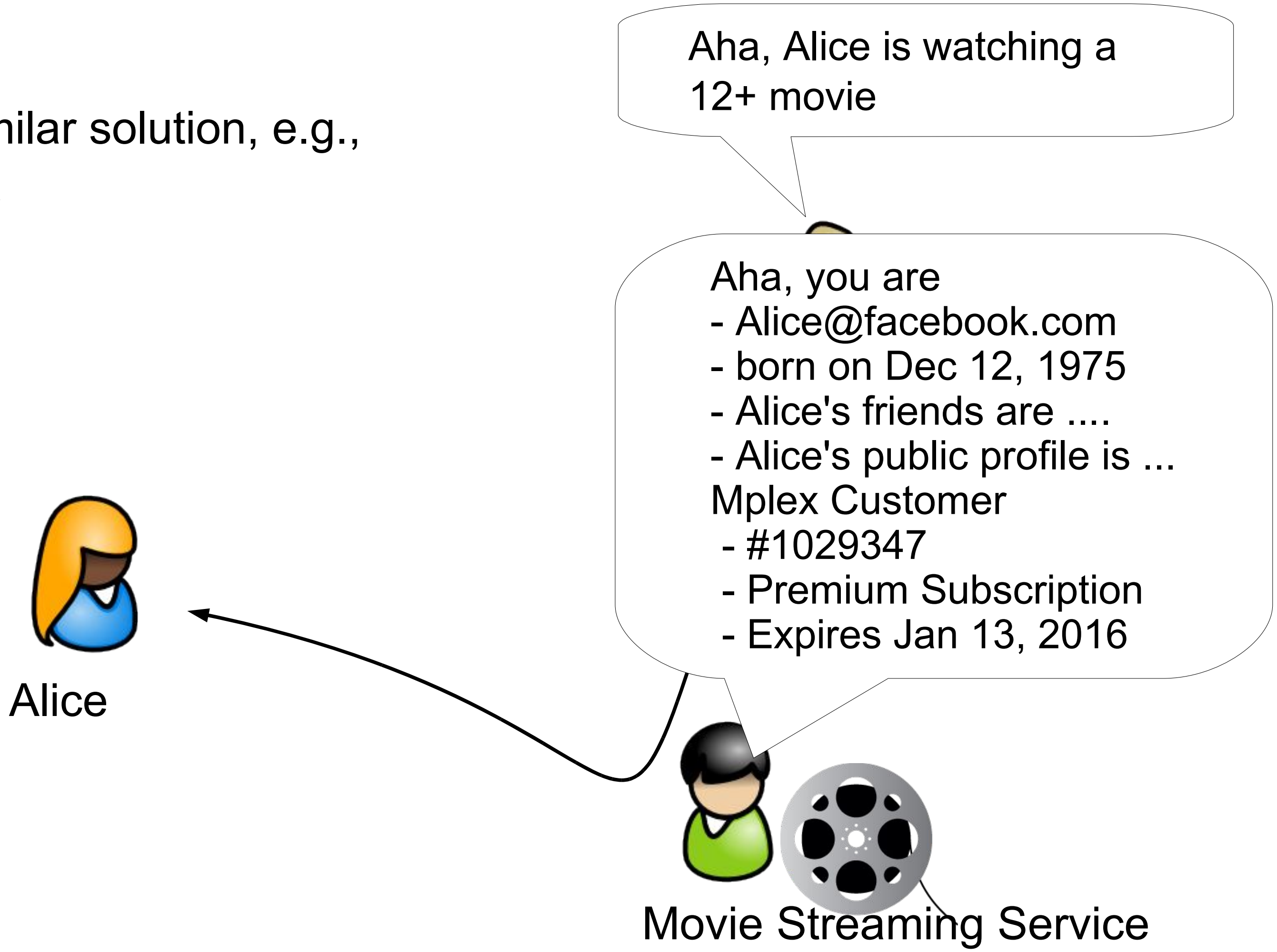
Watching the movie with the traditional solution

With OpenID and similar solution, e.g.,
log-in with Facebook



Watching the movie with the traditional solution

With OpenID and similar solution, e.g., log-in with Facebook



Identity Mixer solves this.

When Alice authenticates to the Movie Streaming Service with Identity Mixer, all the services learns is that Alice

has a subscription

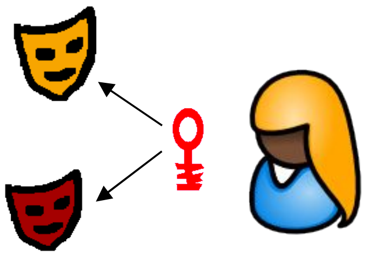
is older than 12

and no more.

Privacy-protecting authentication with IBM Identity Mixer

Like PKI, but better:

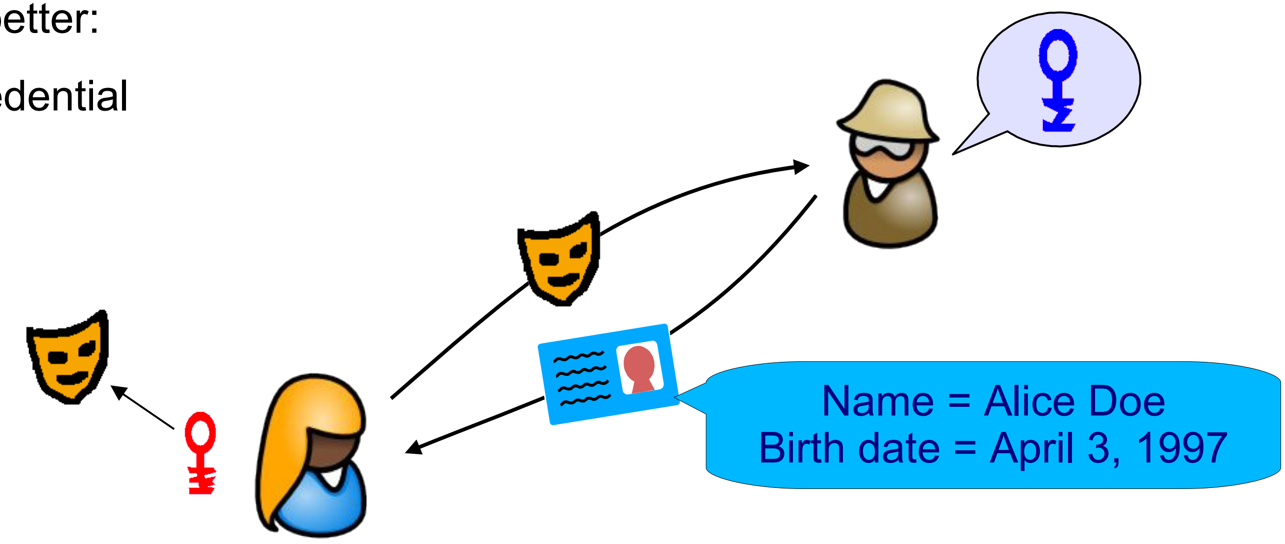
- One secret Identity (secret key)
- Many Public Pseudonyms (public keys)



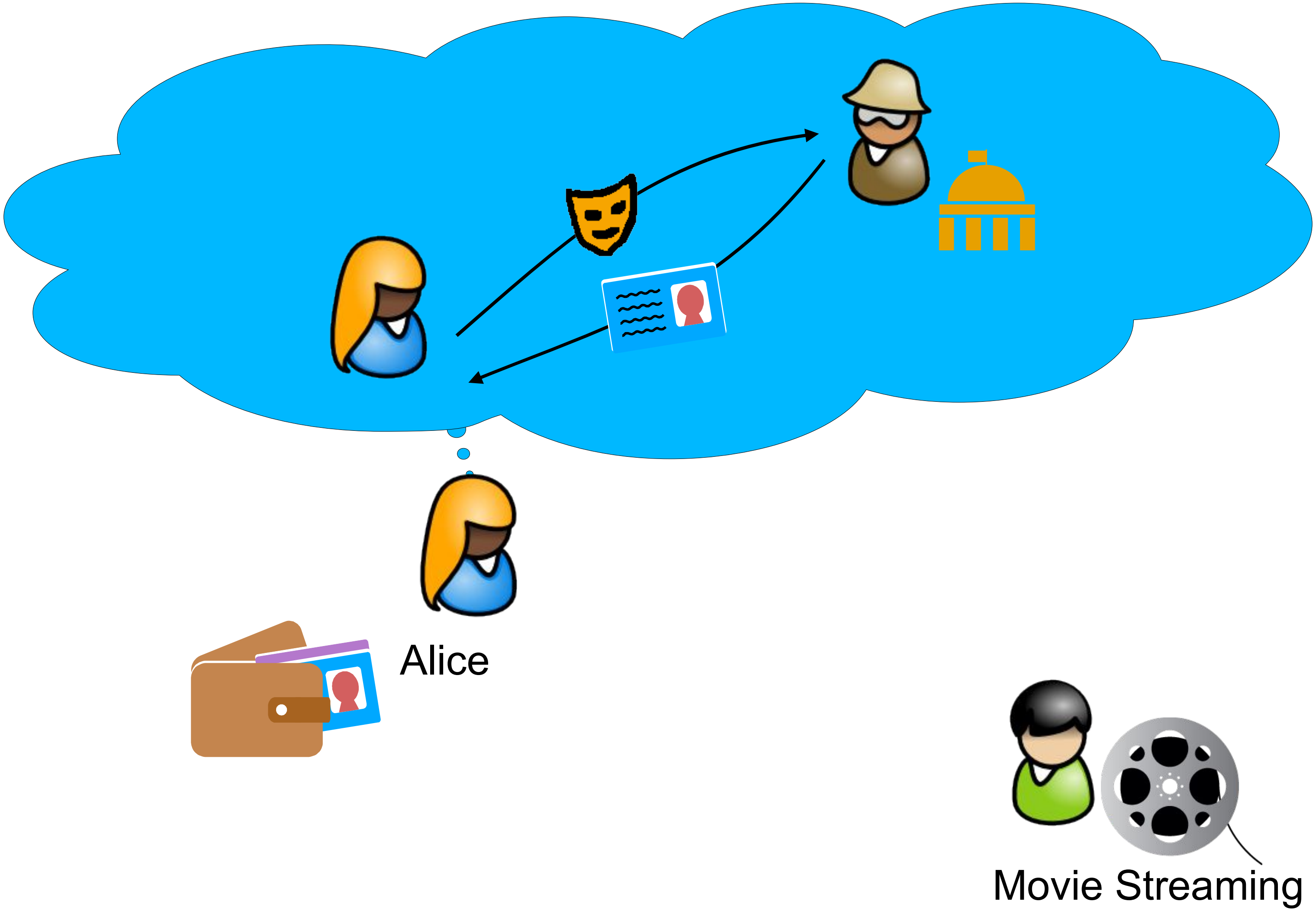
Privacy-protecting authentication with IBM Identity Mixer

Like PKI, but better:

- Issuing a credential

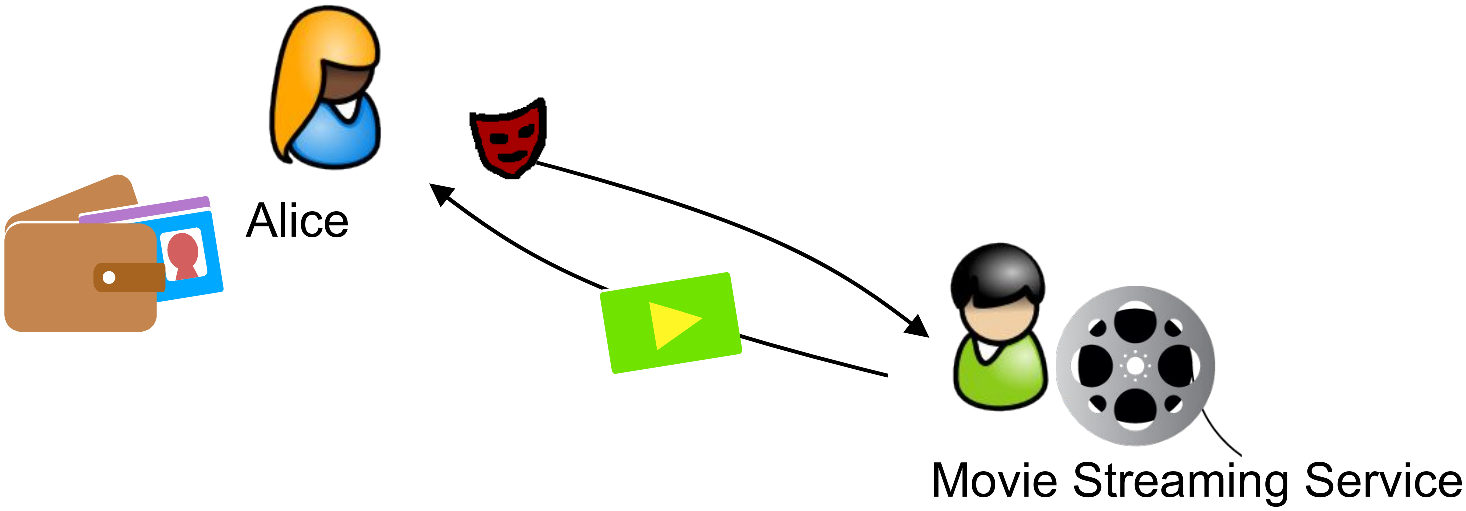


Privacy-protecting authentication with Privacy ABCs




I wish to see
Alice in Wonderland
Alice
You need:
- subscription
- be older than 12
Movie Streaming Service

Privacy-protecting authentication with IBM Identity Mixer

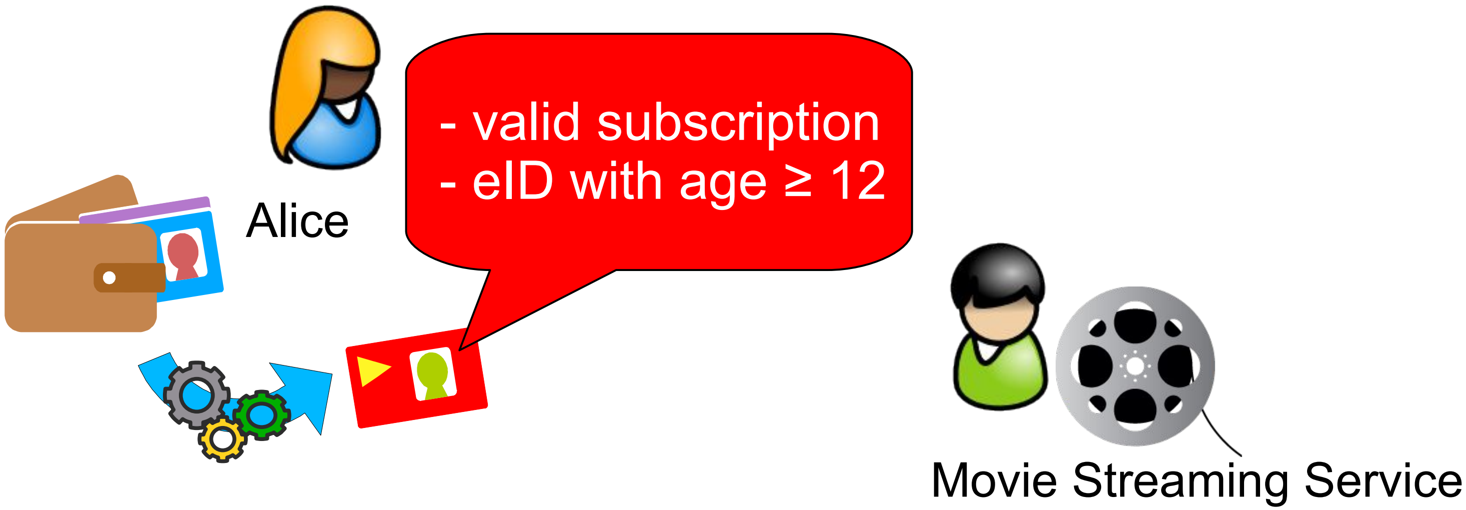


Privacy-protecting authentication with IBM Identity Mixer

Privacy-protecting authentication with IBM Identity Mixer

Like PKI

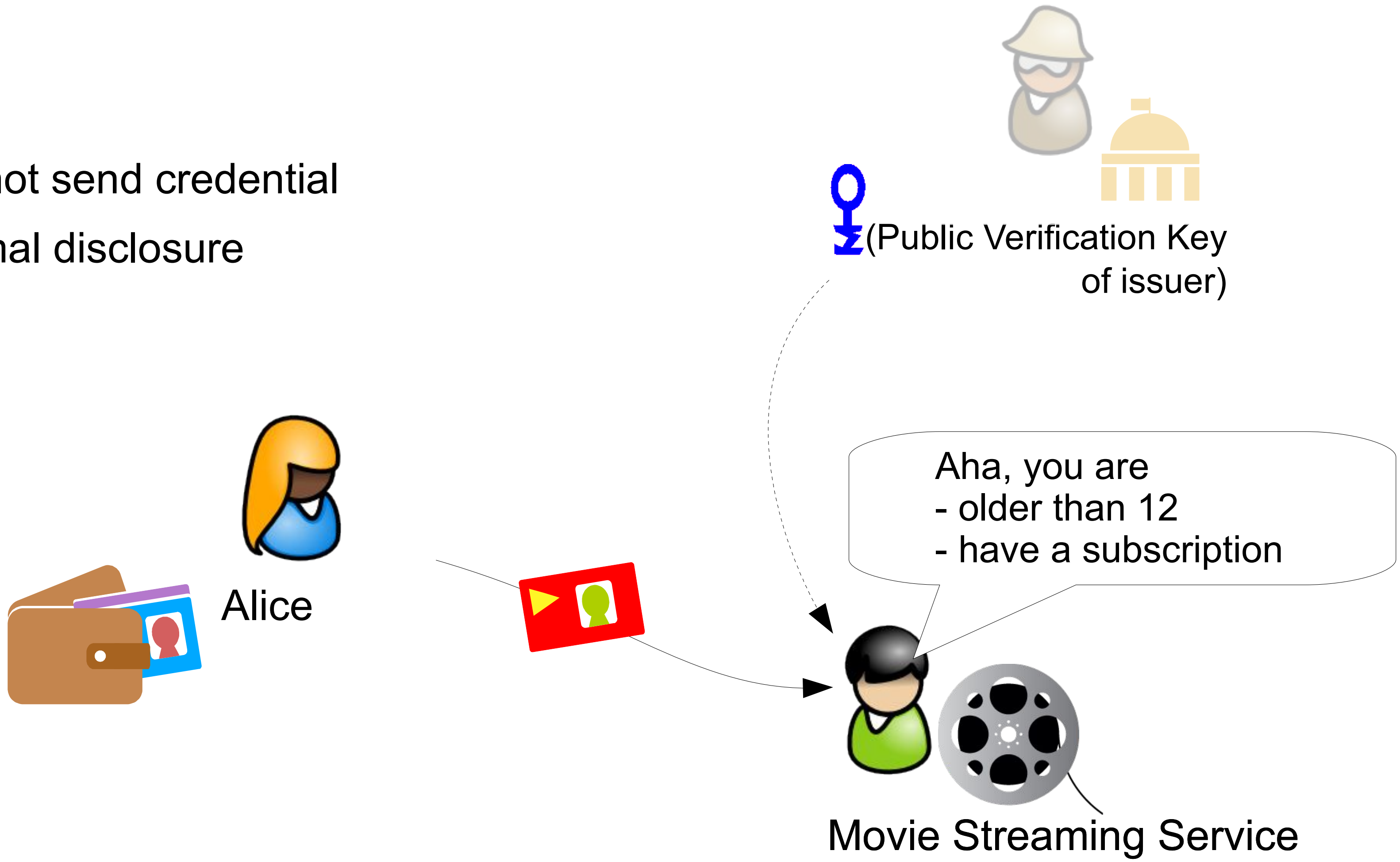
- but does not send credential
- only minimal disclosure



Privacy-protecting authentication with IBM Identity Mixer

Like PKI

- but does not send credential
- only minimal disclosure

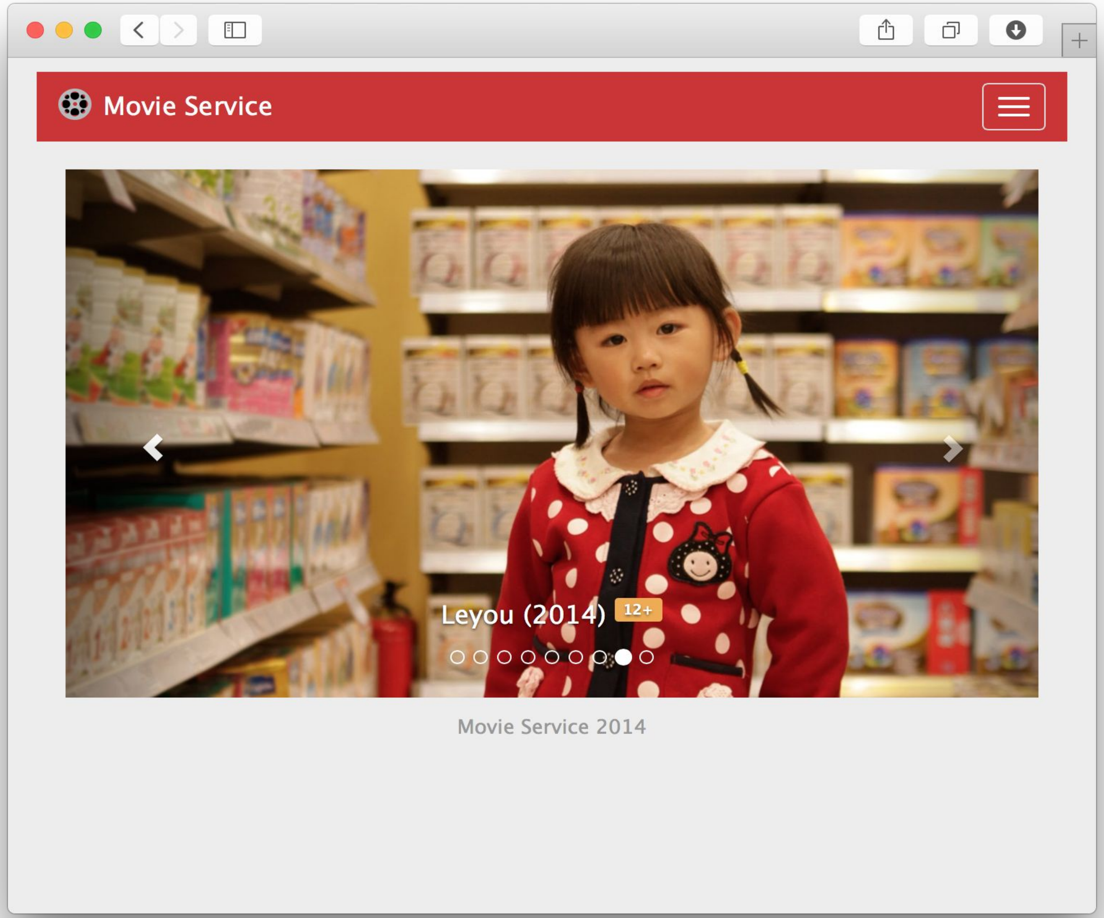


Advantages of Identity Mixer

- **For Users: privacy**
  - minimizing disclosure of personal data
  - keeping their identities safe
  - pseudonymous/anonymous access

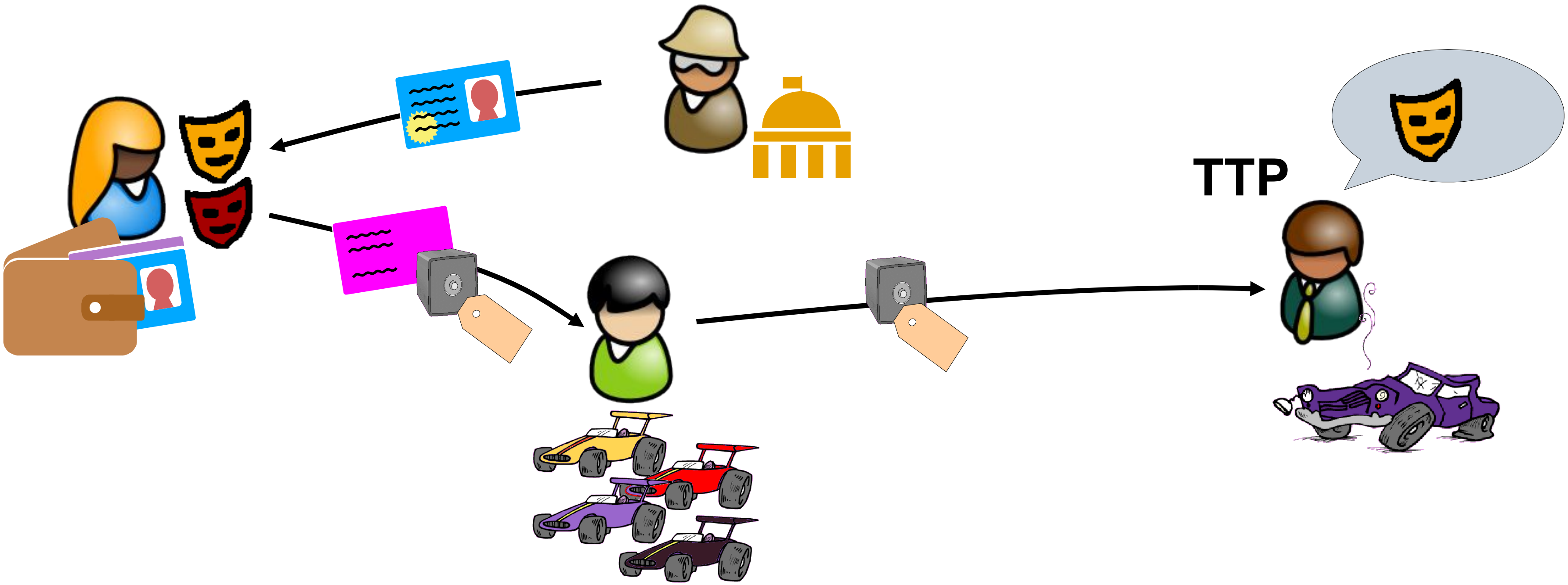
- **For Service Providers: security, accountability, and compliance**
  - avoiding the risk of loosing personal data if it gets stolen
  - compliance with legislation (access control rules, personal data protection)
  - strong authentication (cryptographic proofs replace usernames/passwords)
  - user identification if required (under certain circumstances)

Demo



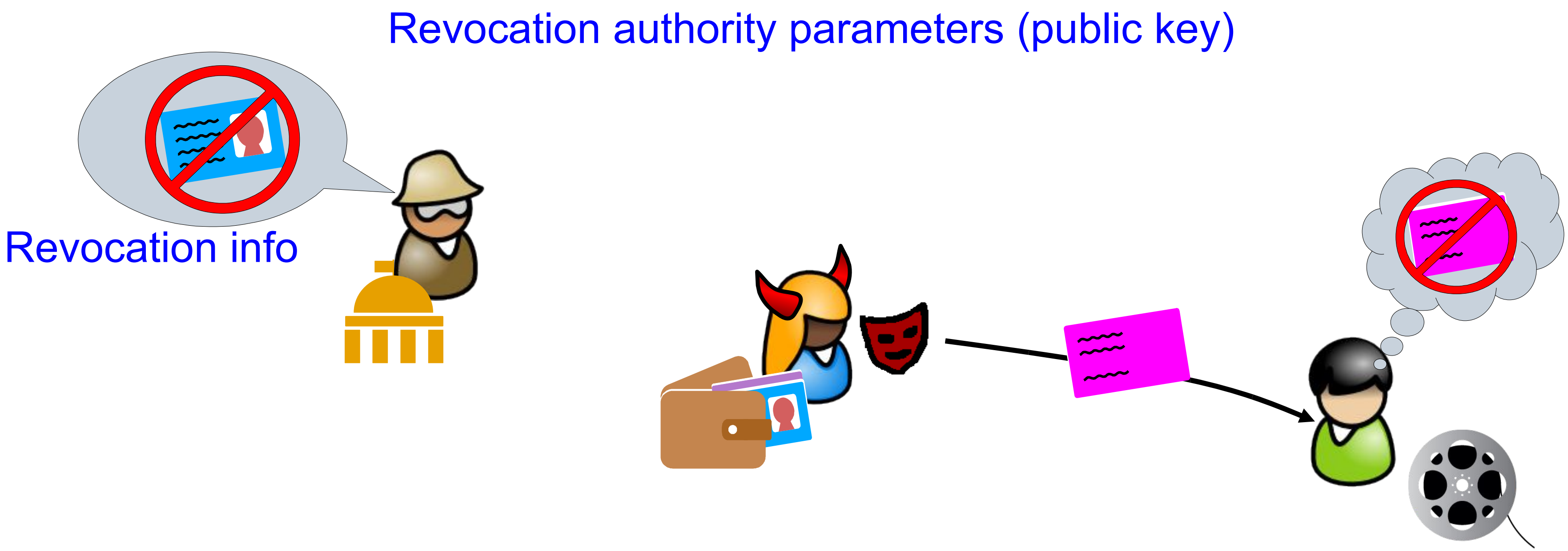
Try yourself at www.ibm.biz/identitymixer on Privacy Day (January 28)

Further Concepts



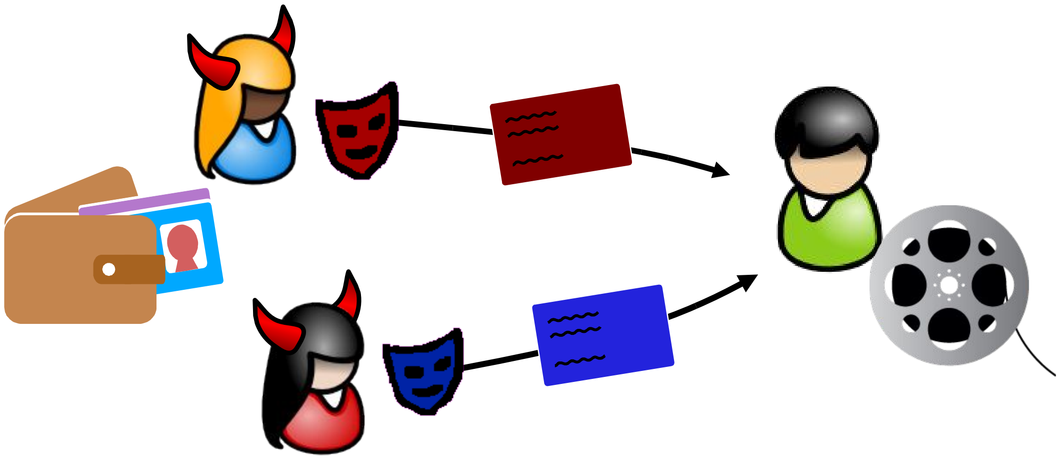
- If car is damaged: ID with insurance or gov't needs be retrieved
- Similarly: verifiably encrypt any certified attribute *(optional)*
- TTP is off-line & can be distributed to lessen trust

Concept – Revocation



- If Alice was speeding, license needs to be revoked!
- There are many different use cases and many solutions
  - Variants of CRL work (using crypto to maintain anonymity)
    - Accumulators
    - Signing entries & Proof, ...
  - Limited validity – certs need to be updated
  - ... For proving age, a revoked driver's license still works

Concept – Usage Limitation



Degree of anonymity can be limited:

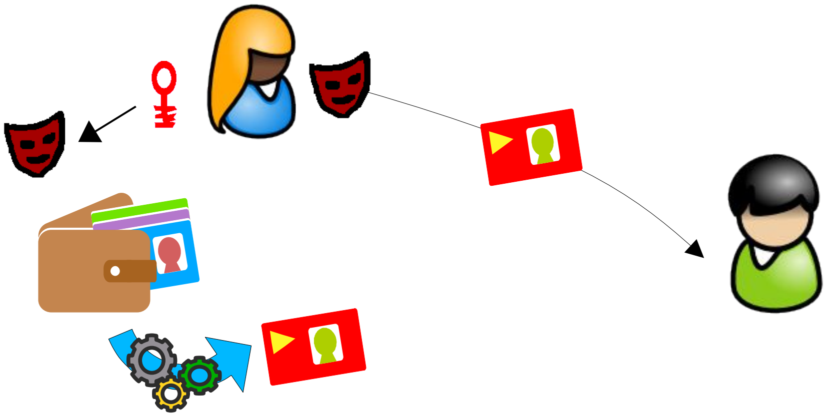
- If Alice and Eve are on-line at the same time, they are caught!
- Use Limitation – anonymous until:
  - If Alice used certs > 100 times total...
  - ... or > 10'000 times with Bob
- Alice's cert can be bound to hardware token (e.g., TPM)

A couple of use cases

Age verification

Proving 12+, 18+, 21+ without disclosing the exact date of birth – privacy and compliance with age-related legislation

- Movie streaming services
- Gaming industry
- Online gambling platforms
- Dating websites
- Social benefits for young/old people



Healthcare

Anonymous treatment of patients (while enabling access control and payments)

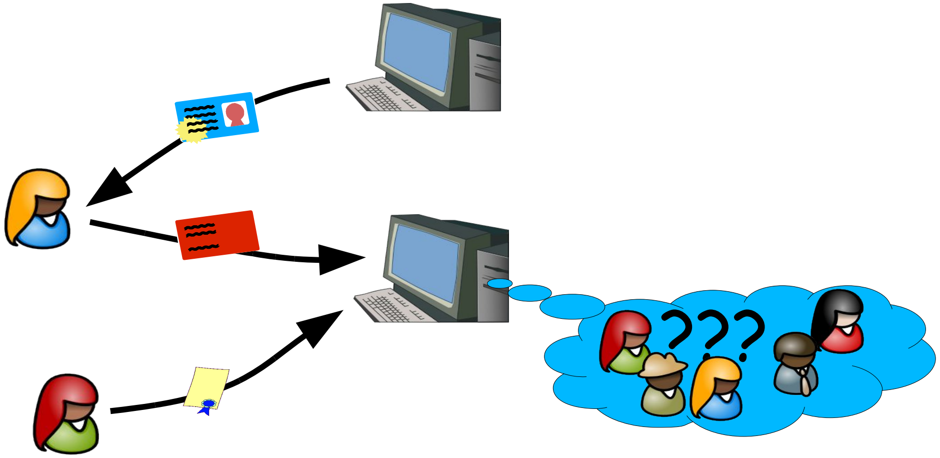
- Anonymous access to patients' records
  - accessing medical test results

- Anonymous consultations with specialists
  - online chat with a psychologist
  - online consultation with IBM Watson
- Eligibility for the premium health insurance
  - proving that the body mass index (BMI) is in the certain range without disclosing the exact weight, height, or BMI

Subscriptions, membership

Who accesses *which data* at which time can reveal sensitive information about the users (their research strategy, location, habits, etc.)

- Patent databases
- DNA databases
- News/Journals/Magazines
- Transportation: tickets, toll roads
- Loyalty programs



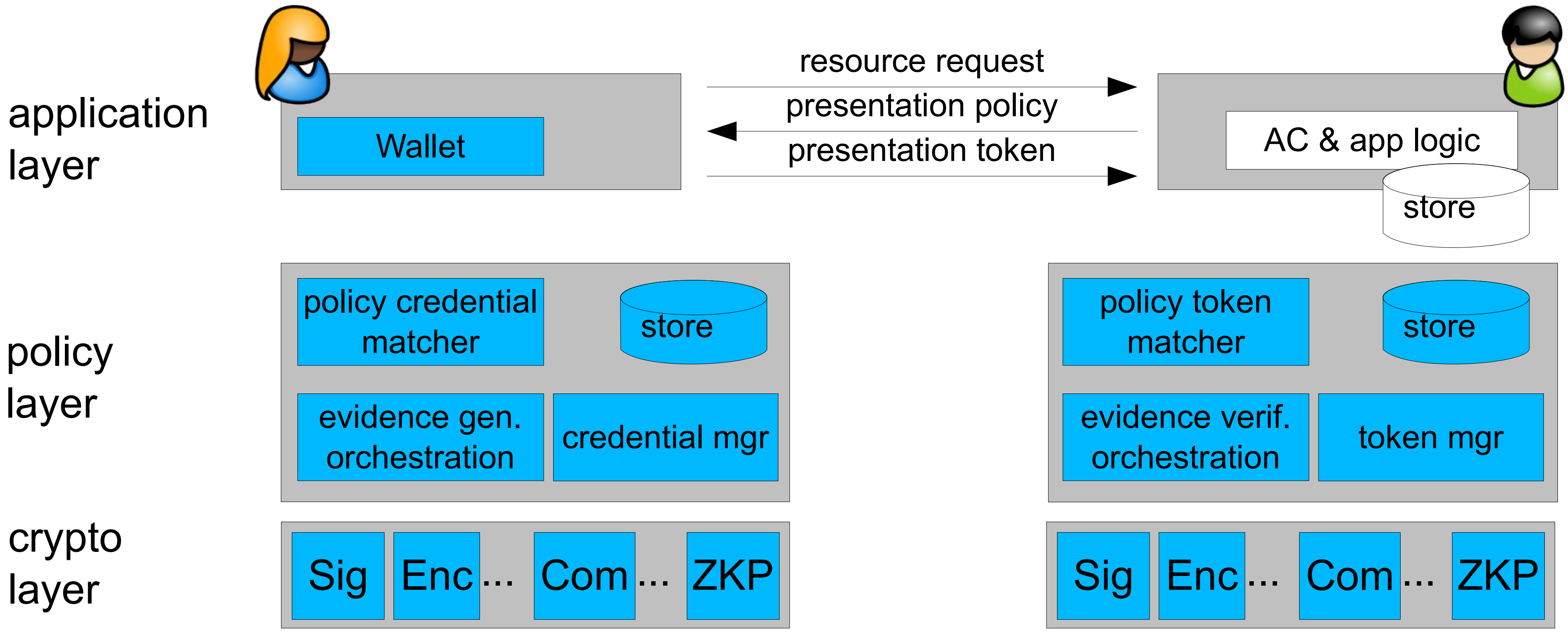
Polls, recommendation platforms

Providing anonymous, but at the same time legitimate feedback

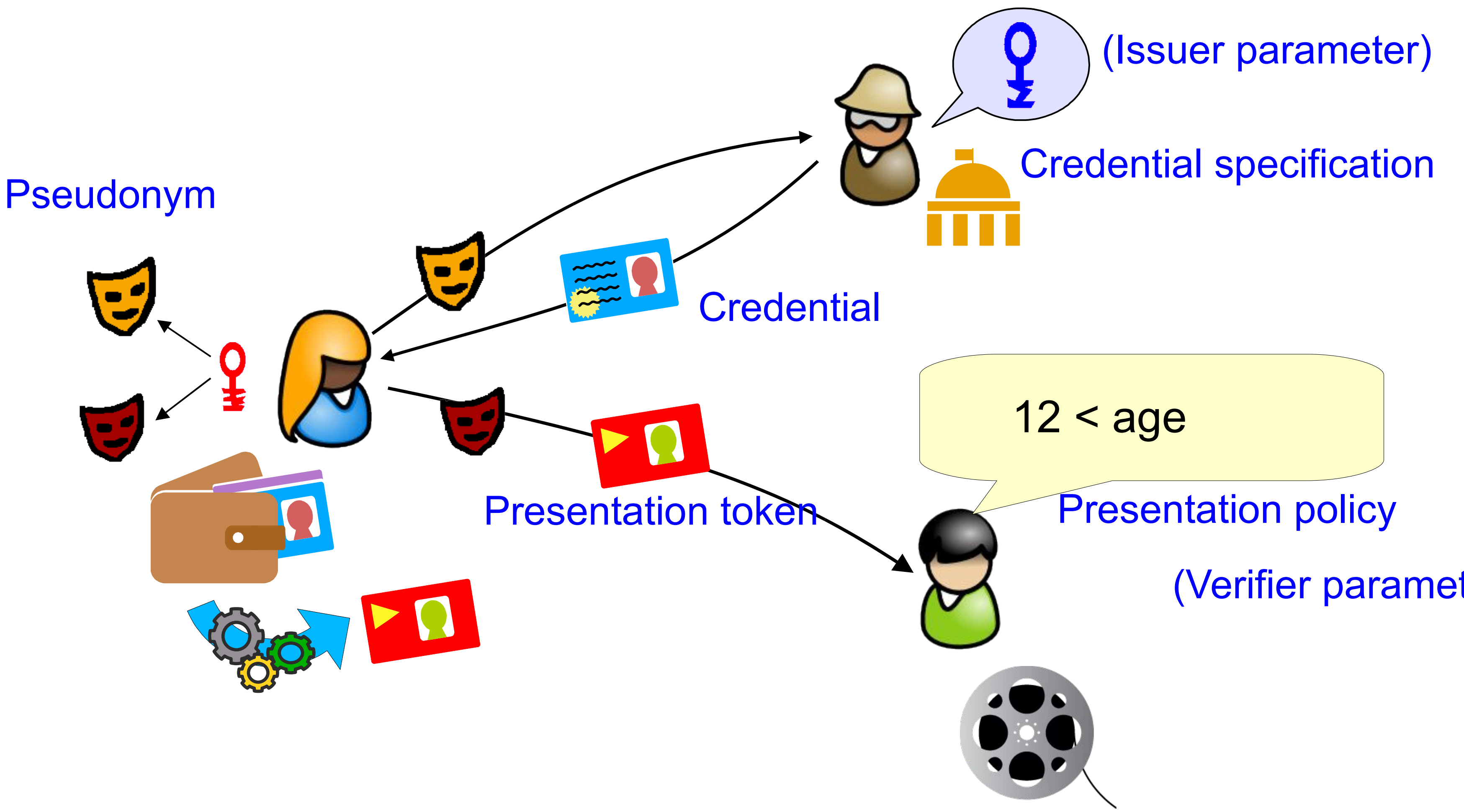
- Online polls
  - applying different restrictions on the poll participants: location, citizenship
- Rating and feedback platforms
  - anonymous feedback for a course only from the students who attended it
  - wikis
  - recommendation platforms

Towards Realizing Anonymous Creds

An Software Stack View on Identity Mixer

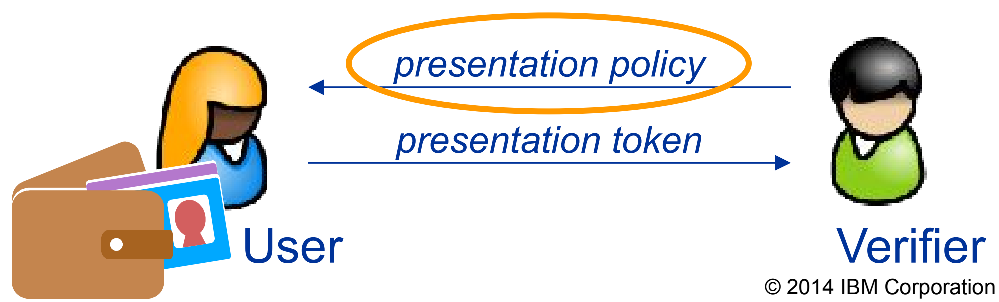


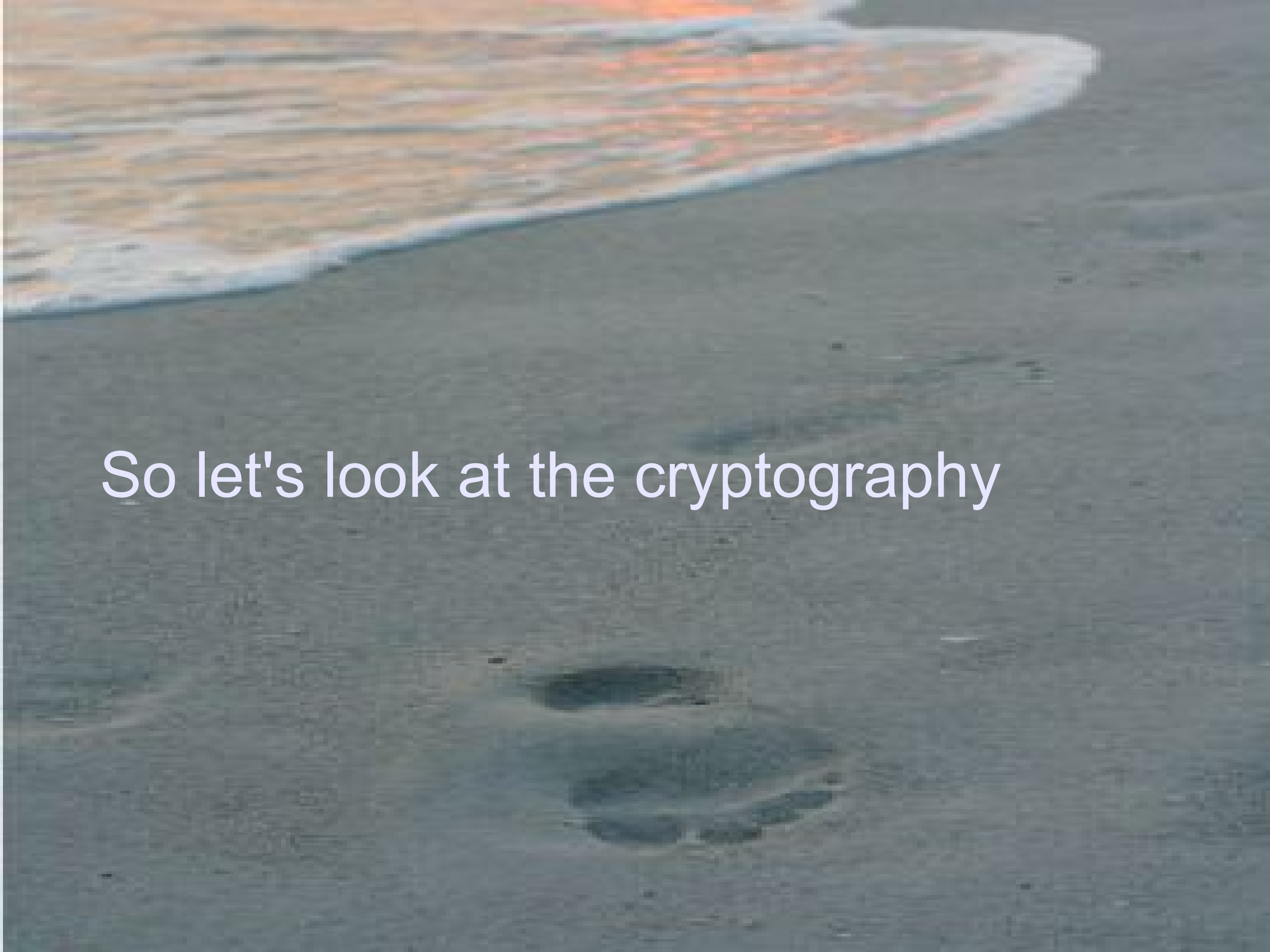
Privacy-protecting authentication with Privacy ABCs



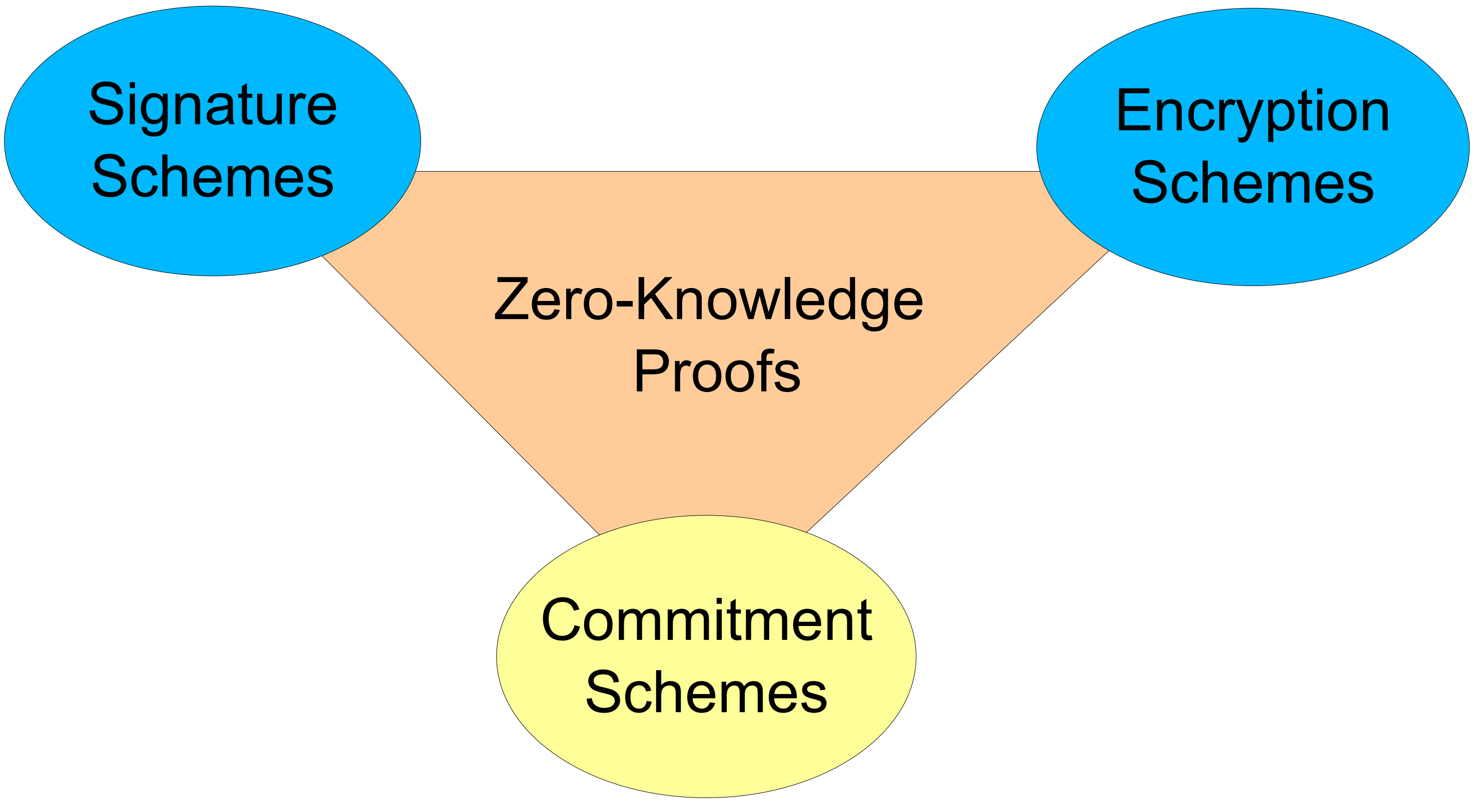
The Policy Layer – An Example: Presentation policy

```xml
<abc:PresentationPolicy PolicyUID="https://movies...com/presentationpolicies/movie1">

   <abc:Message>
     <abc:ApplicationData>  Terms and Conditions </abc:ApplicationData>
   </abc:Message>

   <abc:Credential Alias="#voucher">
     <abc:CredentialSpecAlternatives>
       <abc:CredentialSpecUID>https://movies.....com/specifications/voucher</abc:CredentialSpecUID>
     </abc:CredentialSpecAlternatives>
     <abc:IssuerAlternatives>
       <abc:IssuerParametersUID>https://movies....com/parameters/voucher</abc:IssuerParametersUID>
     </abc:IssuerAlternatives>
   </abc:Credential>

   <abc:AttributePredicate Function="urn:oasis:names:tc:xacml:1.0:function:dateTime-geq">
     <abc:Attribute CredentialAlias="#voucher" AttributeType="Expires" />
     <abc:ConstantValue>2014-06-17T14:06:00Z</abc:ConstantValue>
   </abc:AttributePredicate>

</abc:PresentationPolicy>
```




So let's look at the cryptography

Required Technologies

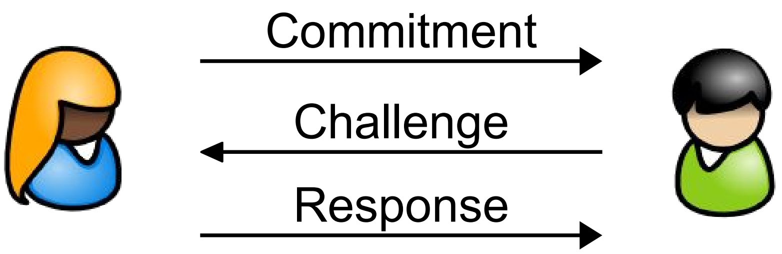


..... challenge is to do all this efficiently!

zero-knowledge proofs

Zero-Knowledge Proofs

- interactive proof between a prover and a verifier about the prover's knowledge



- properties:

  zero-knowledge

    verifier learns nothing about the prover's secret

  proof of knowledge (soundness)

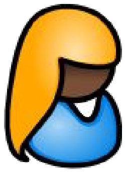
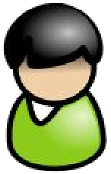
    prover can convince verifier only if she knows the secret

  completeness

    if prover knows the secret she can always convince the verifier

Given group $\langle g \rangle$ and element $y \in \langle g \rangle$.

Prover wants to convince verifier that she *knows* x s.t. $y = g^x$
such that verifier only learns y and g.

x  Prover: | Verifier:  y, g

random r

$t := g^r$ —— t ——→

random c

←—— c ——

$s := r - cx$ —— s ——→ $t = g^s y^c$?

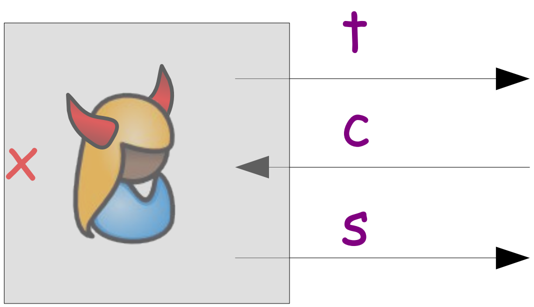
notation: PK{(α): $y = g^\alpha$ }

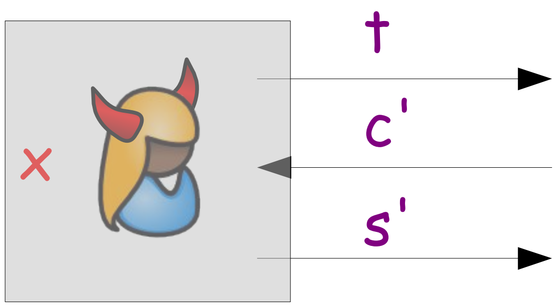
Proof of knowledge: if a prover can successfully convince a verifier, then the secret need to be extractable.

Prover might do protocol computation in any way it wants & we cannot analyse code.

Thought experiment:

- Assume we have prover as a black box → we can reset and rerun prover
- Need to show how secret can be extracted via protocol interface





$$t = g^{s} y^{c} = g^{s'} y^{c'} \quad \rightarrow \quad y^{c'-c} = g^{s-s'}$$

$$\rightarrow \quad y = g^{(s-s')/(c'-c)}$$

$$\rightarrow \quad x = (s-s')/(c'-c) \mod q$$

Zero-knowledge property:

If verifier does not learn anything (except the fact that Alice knows $x = \log_g y$)

Idea: One can simulate whatever Bob “sees”.

Choose random c', s'

compute $t := g^{s'} y^{c'}$

if $c = c'$ send $s' = s$,
otherwise restart

Problem: if domain of c too large, success probability becomes too small

One way to modify protocol to get large domain c:

x Prover: ——— Verifier: y, g

Verifier: random c, v; $h := H(c,v)$

Prover ← h ← Verifier

Prover: random r; $t := g^r$

Prover → t → Verifier

Prover ← c,v ← Verifier

Prover: $h = H(c,v)$?; $s := r - cx$

Prover → s → Verifier

Verifier: $t = g^s y^c$?

notation: $PK\{(\alpha): \; y = g^\alpha\}$

One way to modify protocol to get large domain c:

Choose random c', s'

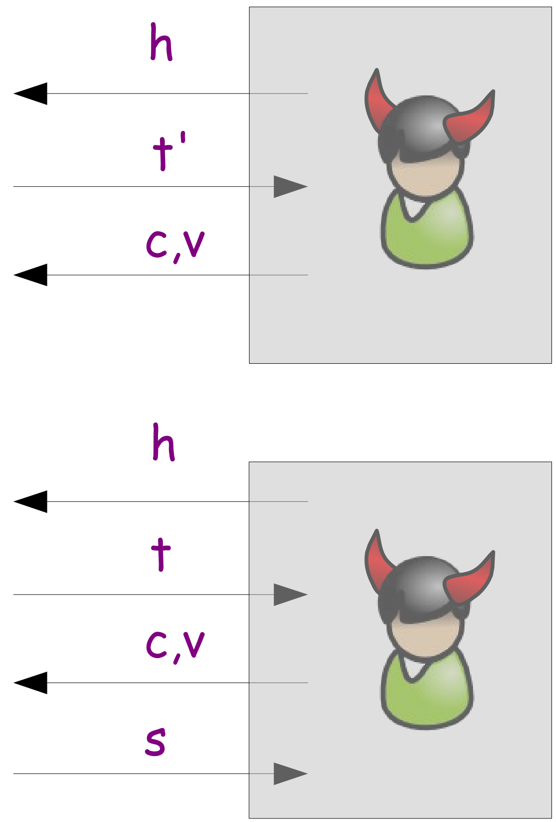
compute $t' := g^{s'} y^{c'}$

after having received c
"reboot" verifier

Choose random s

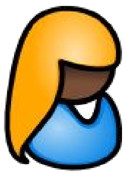
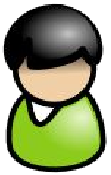
compute $t := g^s y^c$

send s



Given group $\langle g \rangle$ and element $y \in \langle g \rangle$.

Prover wants to convince verifier that she *knows* x s.t. $y = g^x$
such that verifier only learns y and g.

x  Prover: | Verifier:  y, g

random r

$t := g^r$ —— t ——→

random c

←—— c ——

$s := r - cx$ —— s ——→ $t = g^s y^c$?

notation: PK{(α): $y = g^\alpha$ }

Signature $SPK\{(\alpha): y = g^{\alpha}\}(m)$:

Signing a message m:

- chose random $r \in Z_q$ and
- compute $c := H(g^r || m) = H(t || m)$

  $s := r - cx \mod (q)$
- output (c,s)

Verifying a signature (c,s) on a message m:

- check $c = H(g^s y^c || m)$? $\leftrightarrow$ $t = g^s y^c$?

Security:

- underlying protocol is zero-knowledge proof of knowledge
- hash function $H(.)$ behaves as a "random oracle."

Many Exponents:

$$PK\{(\alpha,\beta,\gamma,\delta): \quad y = g^{\alpha} h^{\beta} z^{\gamma} k^{\delta} u^{\beta} \}$$

Logical combinations:

$$PK\{(\alpha,\beta): \quad y = g^{\alpha} \wedge z = g^{\beta} \wedge u = g^{\beta} h^{\alpha} \}$$

$$PK\{(\alpha,\beta): \quad y = g^{\alpha} \vee z = g^{\beta} \}$$

Intervals and groups of different order (under SRSA):

$$PK\{(\alpha): \; y = g^{\alpha} \wedge \alpha \in [A,B] \}$$

$$PK\{(\alpha): \; y = g^{\alpha} \wedge z = g^{\alpha} \wedge \alpha \in [0,\min\{ord(g),ord(g)\}] \}$$

Non-interactive (Fiat-Shamir heuristic, Schnorr Signatures):

$$SPK\{(\alpha): \; y = g^{\alpha} \}(m)$$

Let $g, h, C1, C2, C3$ be group elements.

Now, what does

$PK\{(\alpha 1, \beta 1, \alpha 2, \beta 2, \alpha 3, \beta 3): \quad C1 = g^{\alpha 1} h^{\beta 1} \wedge C2 = g^{\alpha 2} h^{\beta 2} \wedge C3 = g^{\alpha 3} h^{\beta 3} \wedge C3 = g^{\alpha 1} g^{\alpha 2} h^{\beta 3}\}$

mean?

→ Prover knows values $\alpha 1, \beta 1, \alpha 2, \beta 2, \beta 3$ such that

$C1 = g^{\alpha 1} h^{\beta 1}$, $C2 = g^{\alpha 2} h^{\beta 2}$ and

$C3 = g^{\alpha 1} g^{\alpha 2} h^{\beta 3} = g^{\alpha 1 + \alpha 2} h^{\beta 3} = g^{\alpha 3} h^{\beta 3}$

→ $\alpha 3 = \alpha 1 + \alpha 2 \pmod q$

And what about:

$PK\{(\alpha 1, \ldots, \beta 3): \quad C1 = g^{\alpha 1} h^{\beta 1} \wedge C2 = g^{\alpha 2} h^{\beta 2} \wedge C3 = g^{\alpha 3} h^{\beta 3} \wedge C3 = g^{\alpha 1} (g^5)^{\alpha 2} h^{\beta 3}\}$

→ $C3 = g^{\alpha 1} g^{\alpha 2} h^{\beta 3} = g^{\alpha 1 + 5\alpha 2} h^{\beta 3}$

→ $\alpha 3 = \alpha 1 + 5\alpha 2 \pmod q$

Some Example Proofs and Their Analysis

Let $g, h, C1, C2, C3$ be group elements.

Now, what does

$$PK\{(\alpha 1,\dots,\beta 3): \quad C1 = g^{\alpha 1}h^{\beta 1} \wedge C2 = g^{\alpha 2}h^{\beta 2} \wedge C3 = g^{\alpha 3}h^{\beta 3} \wedge C3 = C2^{\alpha 1}h^{\beta 3}\}$$ mean?

→ Prover knows values $\alpha 1, \beta 1, \alpha 2, \beta 2, \beta 3$ such that

$C1 = g^{\alpha 1}h^{\beta 1}$, $C2 = g^{\alpha 2}h^{\beta 2}$ and

$$C3 = C2^{\alpha 1}h^{\beta 3} = (g^{\alpha 2}h^{\beta 2})^{\alpha 1}h^{\beta 3} = g^{\alpha 2 \cdot \alpha 1}h^{\beta 3 + \beta 2 \cdot \alpha 1}$$

$$C3 = g^{\alpha 2 \cdot \alpha 1}\, h^{\beta 3 + \beta 2 \cdot \alpha 1} = g^{\alpha 3}\, h^{\beta 3'}$$

$$\rightarrow \alpha 3 = \alpha 1 \cdot \alpha 2 \pmod q$$

And what about

$$PK\{(\alpha 1, \beta 1\ \beta 2): \quad C1 = g^{\alpha 1}h^{\beta 1} \wedge C2 = g^{\alpha 2}h^{\beta 2} \wedge C2 = C1^{\alpha 1}h^{\beta 2}\}$$

$$\rightarrow \alpha 2 = \alpha 1^2 \pmod q$$

Let $g, h, C1, C2, C3$ be group elements.

Now, what does

$PK\{(\alpha 1,..,\beta 2): \quad C1= g^{\alpha 1}h^{\beta 1} \wedge C2= g^{\alpha 2}h^{\beta 2} \wedge g = (C2/C1)^{\alpha 1}h^{\beta 2}\}$ mean?

→ Prover knows values $\alpha, \beta 1, \beta 2$ such that

$$C1= g^{\alpha 1}h^{\beta 1}$$

$$g = (C2/C1)^{\alpha 1}h^{\beta 2} = (C2\, g^{-\alpha 1}h^{-\beta 1})^{\alpha 1}\, h^{\beta 2}$$

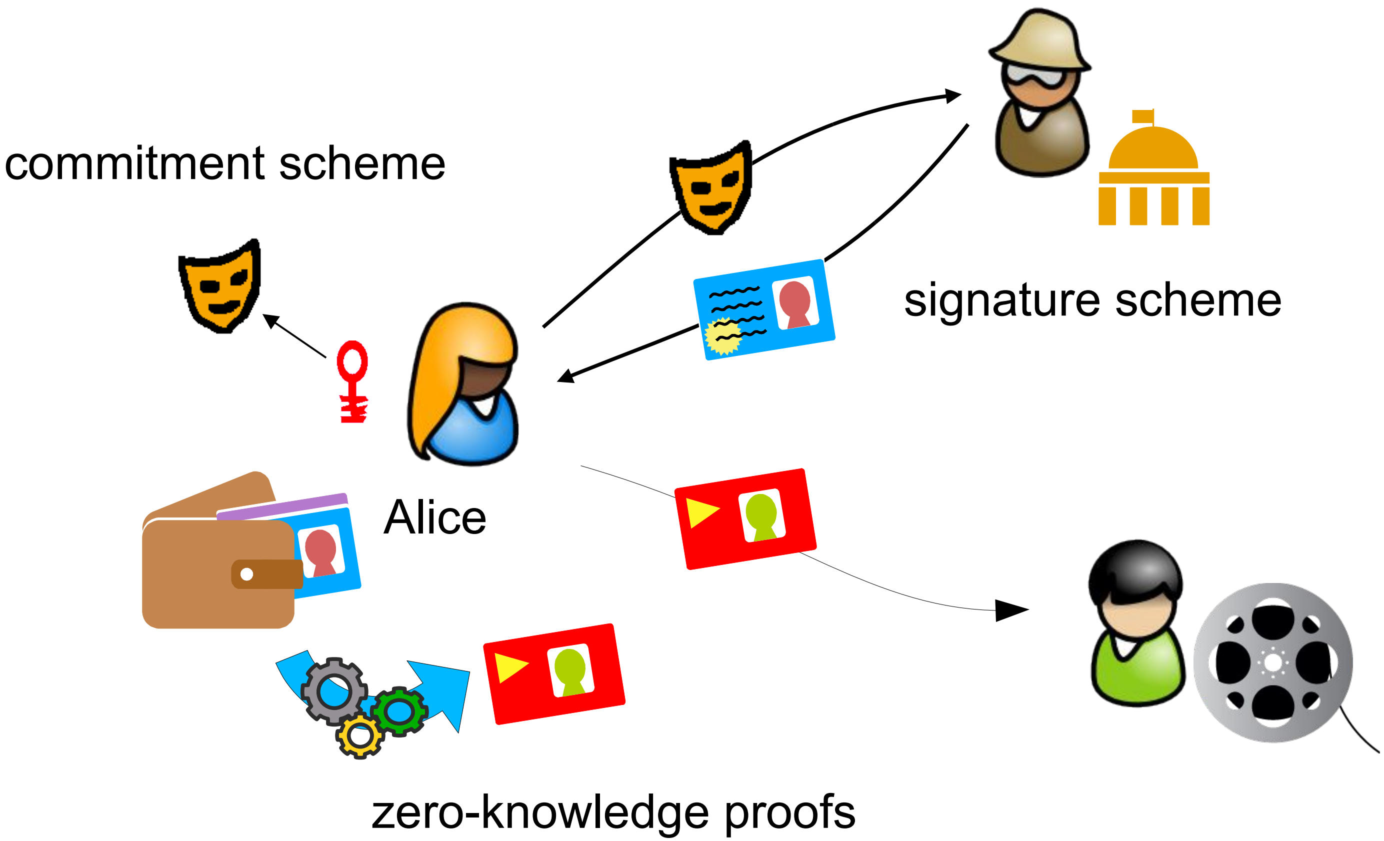
→ $$g^{1/\alpha 1} = C2\, g^{-\alpha 1}h^{-\beta 1}\ h^{\beta 2/\alpha 1}$$

$$C2 = g^{\alpha 1}\, h^{\beta 1}\ h^{-\beta 2/\alpha 1}\, g^{1/\alpha 1} = g^{\alpha 1 + 1/\alpha 1}\, h^{\beta 1-\beta 2/\alpha 1}$$

$$C2 = g^{\alpha 2}\, h^{\beta 2}$$

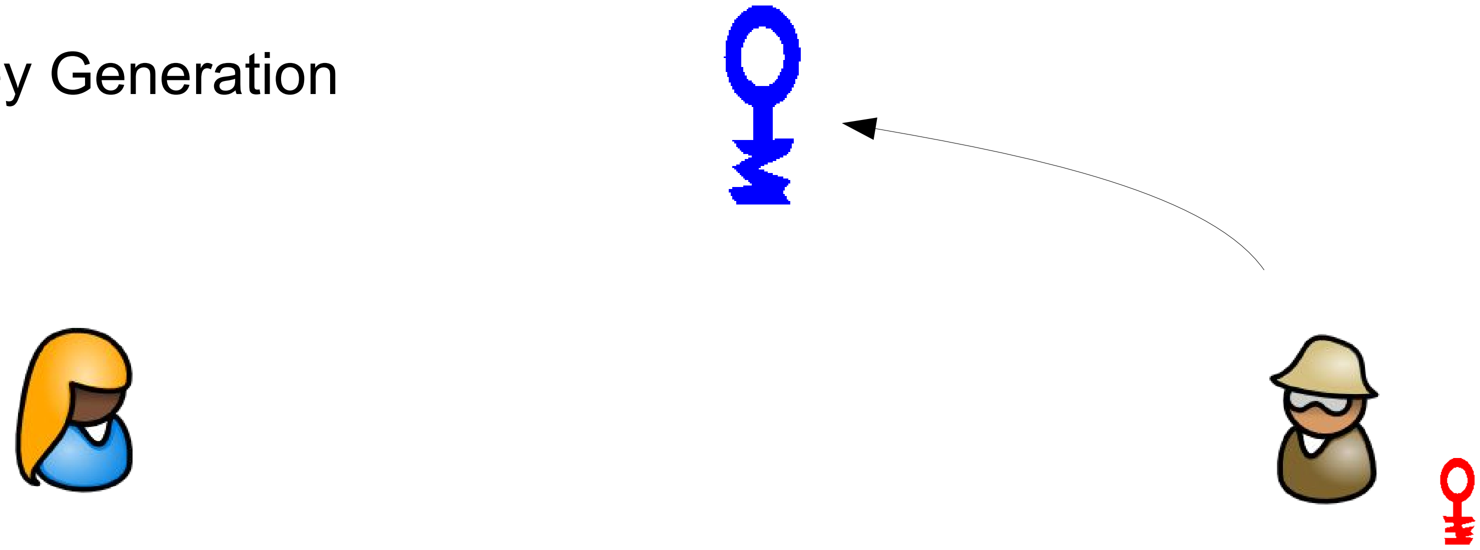
$$\alpha 2 = \alpha 1 + \alpha 1^{-1} \pmod q$$

Privacy-protecting authentication with Privacy ABCs

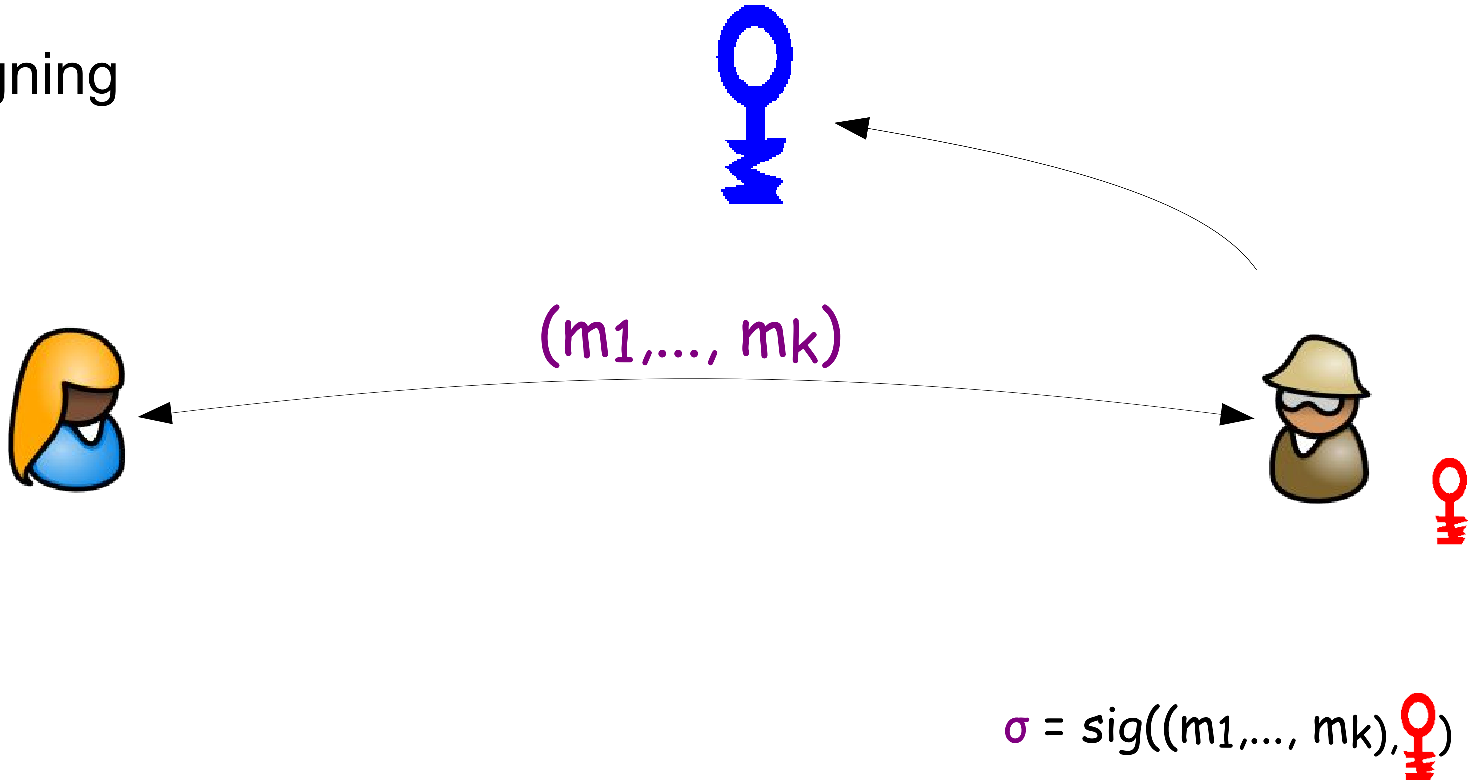


signature schemes

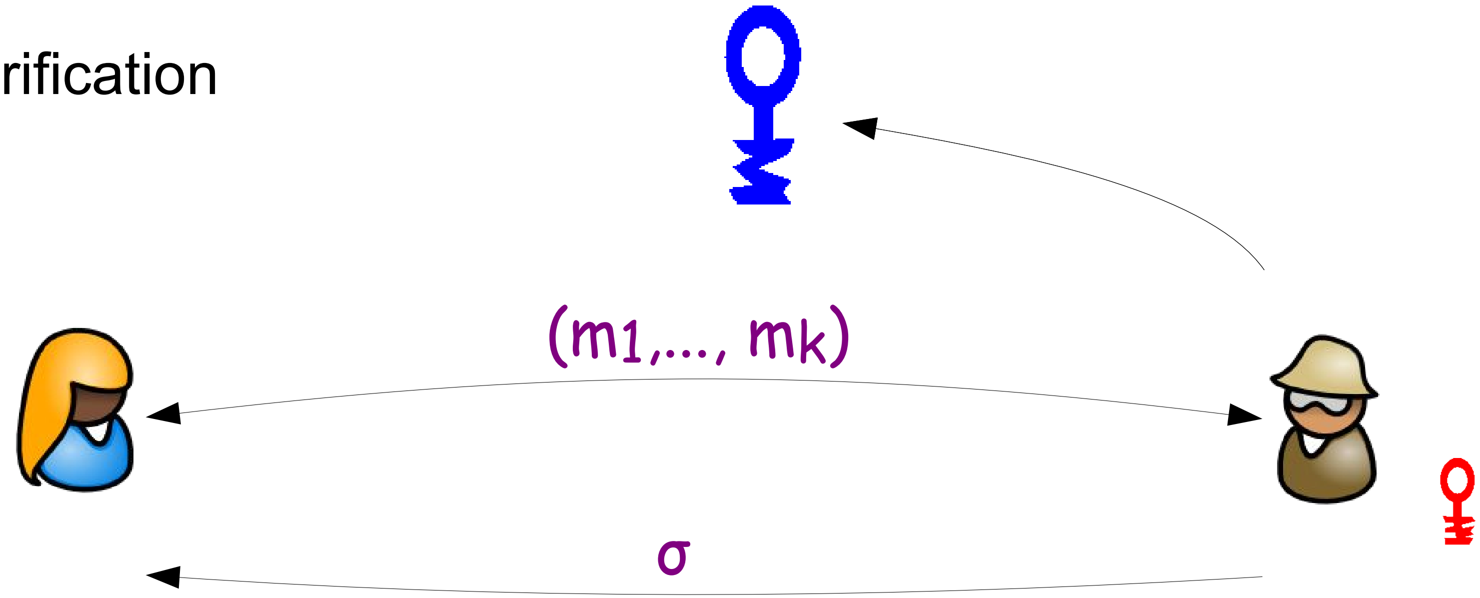
Key Generation



Signing



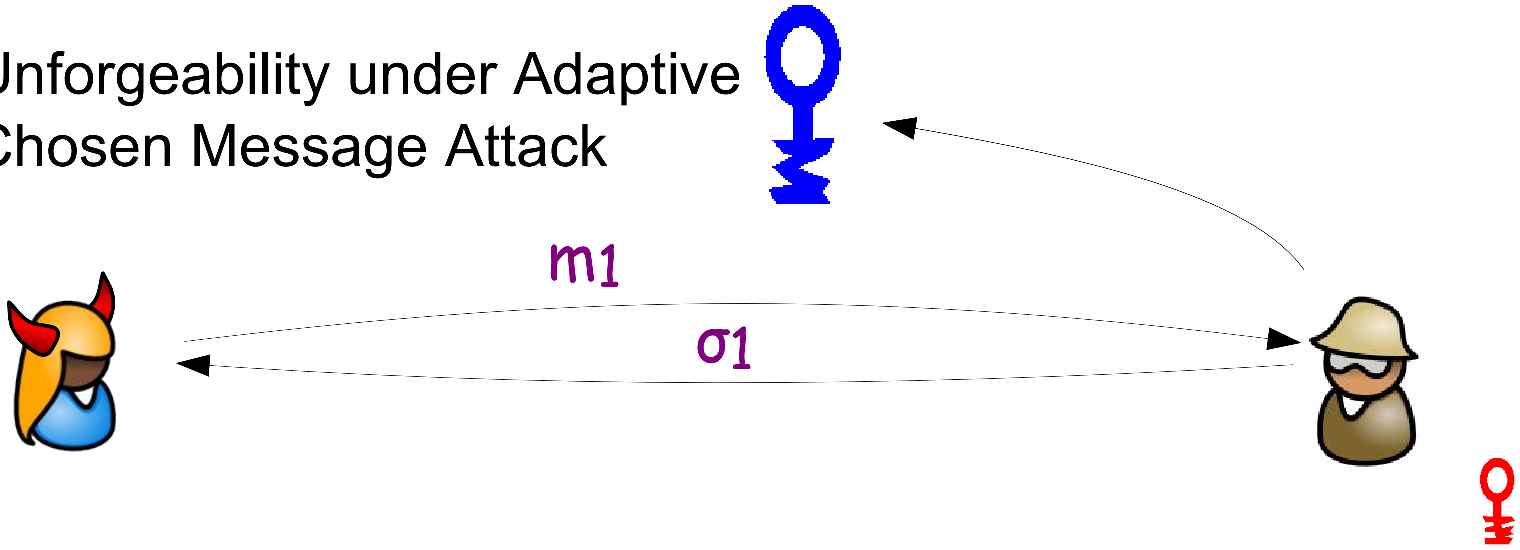
Verification



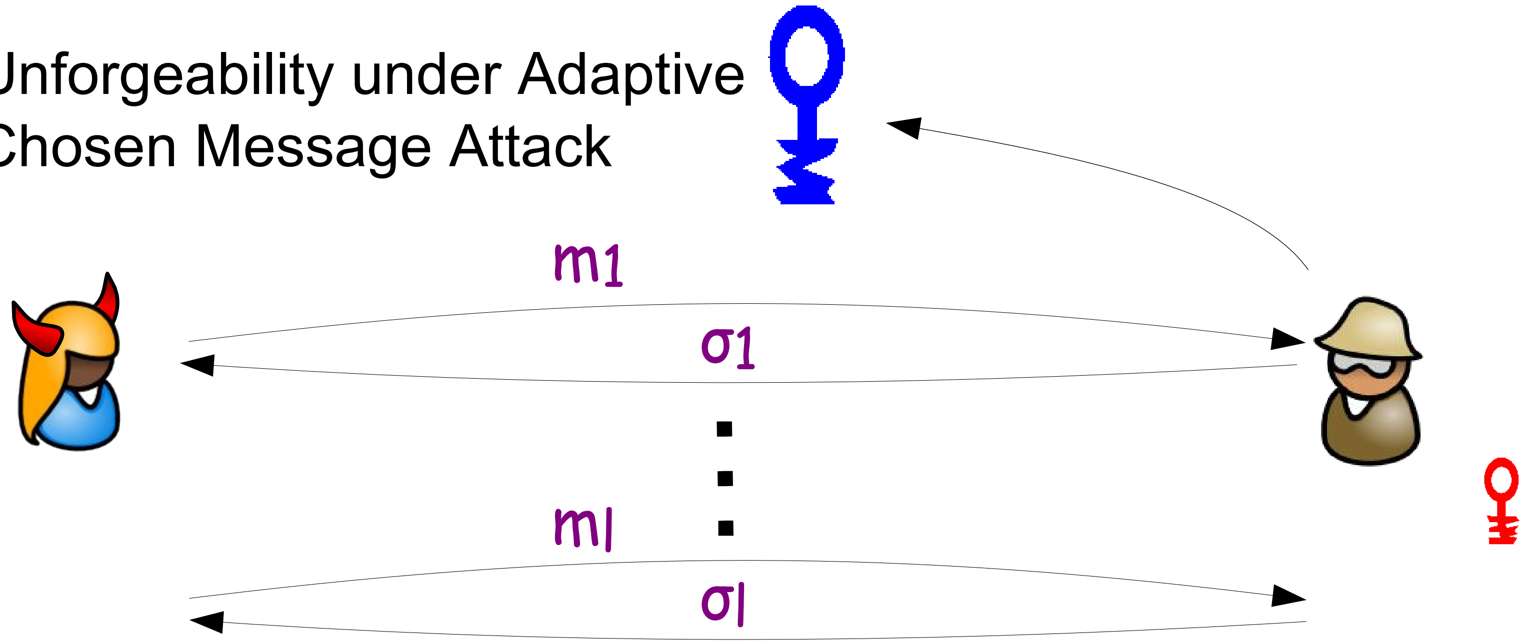
σ = sig((m1,..., mk), [private key])

ver(σ,(m1,..., mk), [public key]) = true

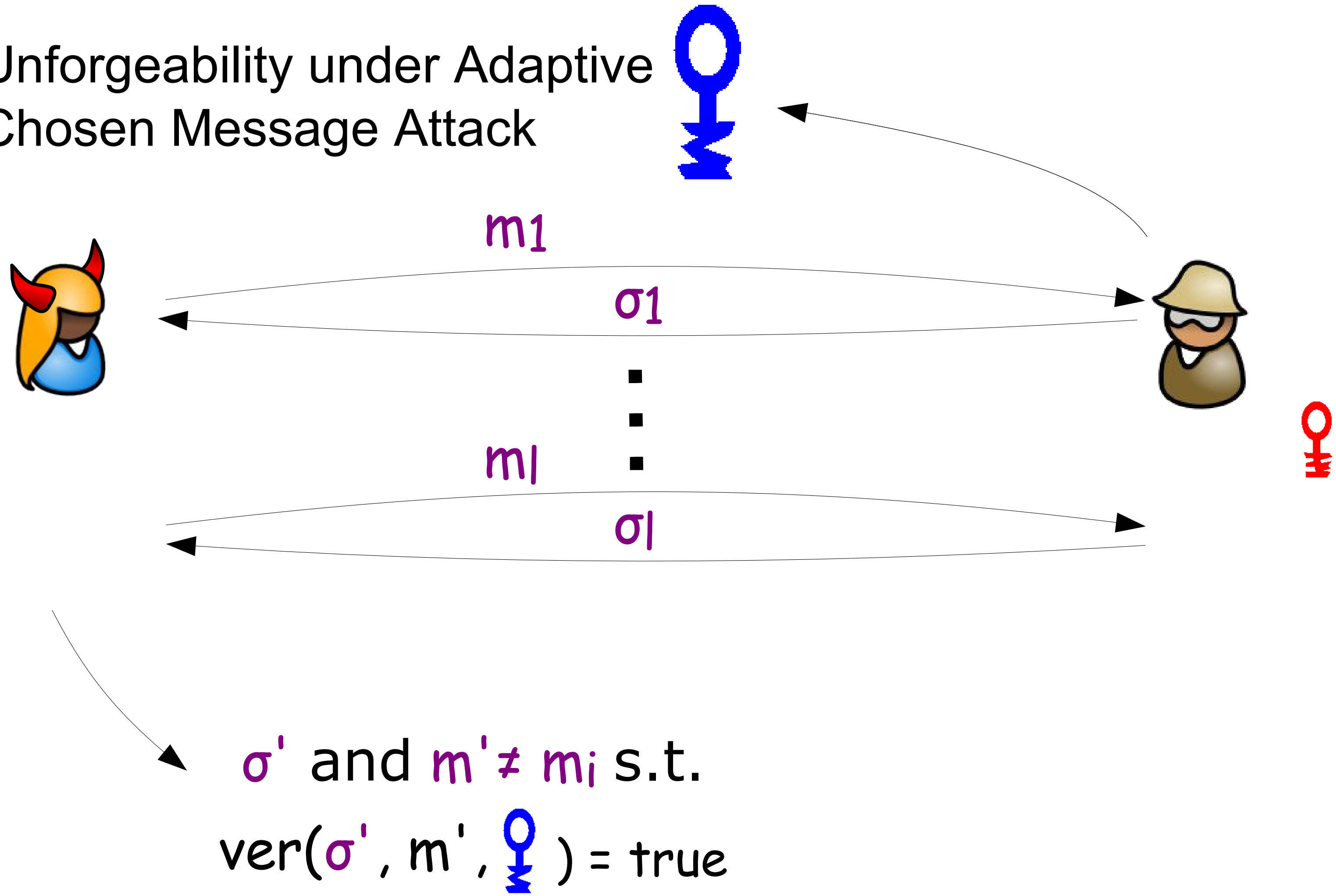
Unforgeability under Adaptive Chosen Message Attack



Unforgeability under Adaptive Chosen Message Attack

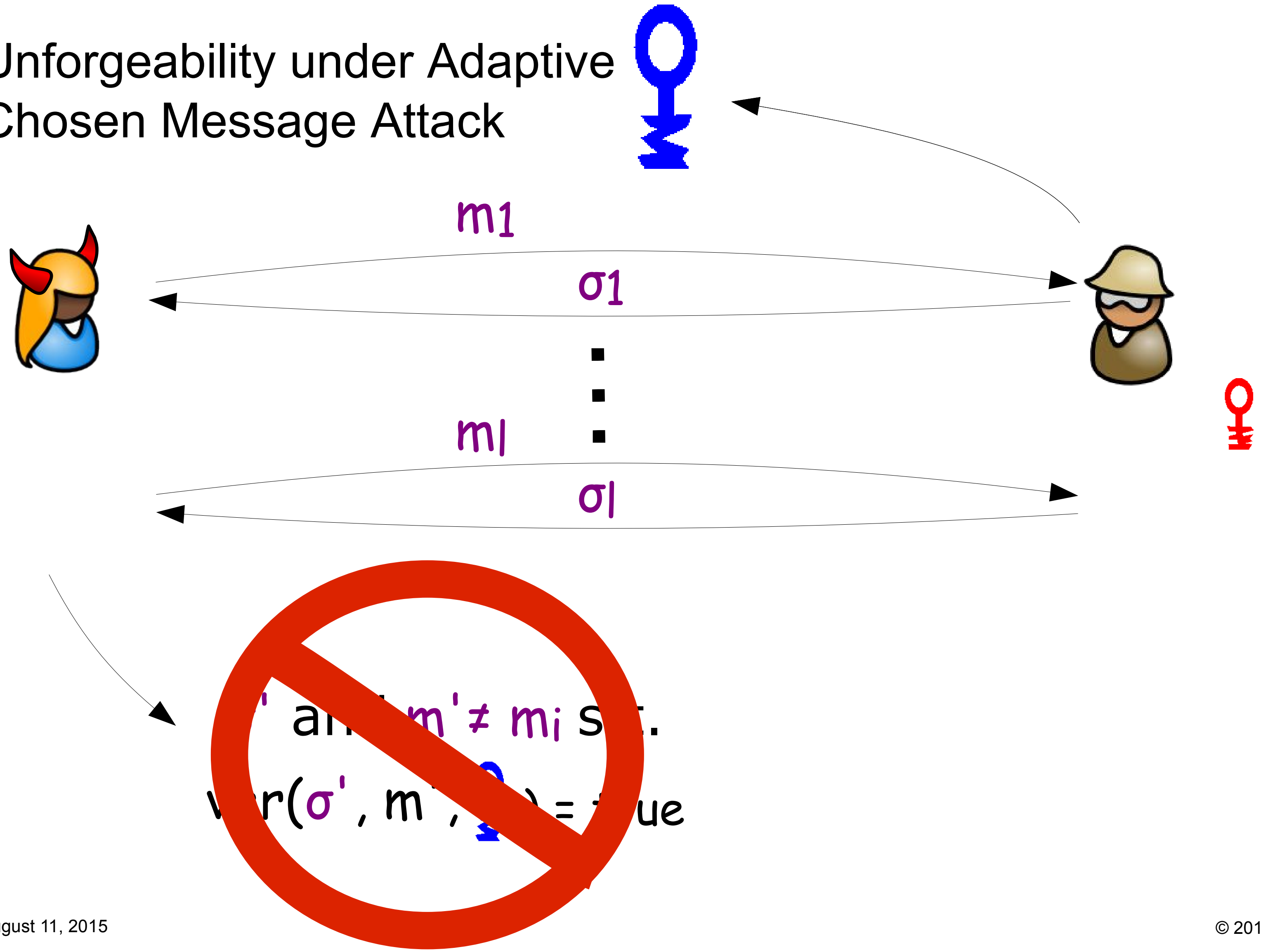


Unforgeability under Adaptive Chosen Message Attack



IBM

Unforgeability under Adaptive Chosen Message Attack



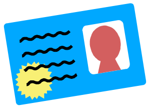
RSA Signature Scheme – for reference

Rivest, Shamir, and Adlemann 1978

Secret Key: two random primes p and q

Public Key: $n := pq$, prime e,
and collision-free hash function

$$H: \{0,1\}^* \rightarrow \{0,1\}^\ell$$



Computing signature on a message $m \in \{0,1\}^*$

$$d := 1/e \bmod (p-1)(q-1)$$

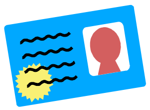
$$s := H(m)^d \bmod n$$

Verification of signature s on a message $m \in \{0,1\}^*$

$$s^e = H(m) \pmod{n}$$

Correctness: $s^e = (H(m)^d)^e = H(m)^{d \cdot e} = H(m) \pmod{n}$

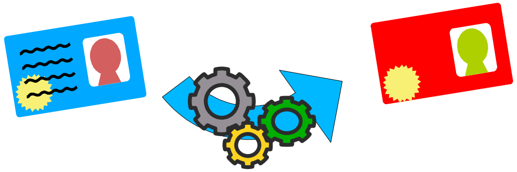
Verification signature on a message $m \in \{0,1\}^*$



$$s^e := H(m) \quad (\text{mod } n)$$

Wanna do proof of knowledge of signature on a message, e.g.,

$$PK\{\, (m,s):\ s^e = H(m)\ (\text{mod } n)\, \}$$



But this is not a valid proof expression!!!! :-(

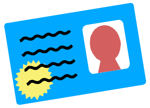
Public key of signer: RSA modulus n and a_i, b, d $\in QR_n$,

Secret key: factors of n

To sign k messages $m1, \ldots, mk \in \{0,1\}^{\ell}$:

- choose random *prime* $2^{\ell+2} > e > 2^{\ell+1}$ and *integer* $s \approx n$
- compute c :

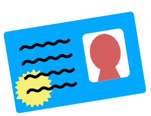
$$c = (d / (a_1^{m1} \cdot \ldots \cdot a_k^{mk} \, b^s))^{1/e} \bmod n$$



- signature is (c,e,s)

To verify a signature (c,e,s) on messages $m1, \ldots, mk$:

- $m1, \ldots, mk \in \{0,1\}^{\ell}$:
- $e > 2^{\ell+1}$
- $d = c^e a_1^{m1} \cdot \ldots \cdot a_k^{mk} b^s \bmod n$

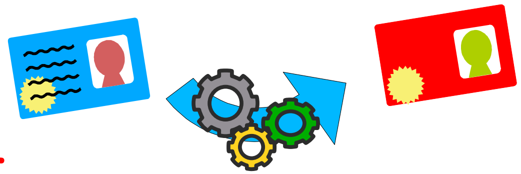


Theorem: *Signature scheme is secure against adaptively chosen message attacks under Strong RSA assumption.*

Proving Knowledge of a CL-signature

Recall: $d = c^{e} \, a1^{m1} a2^{m2} \, b^{s}$ mod n

Observe:

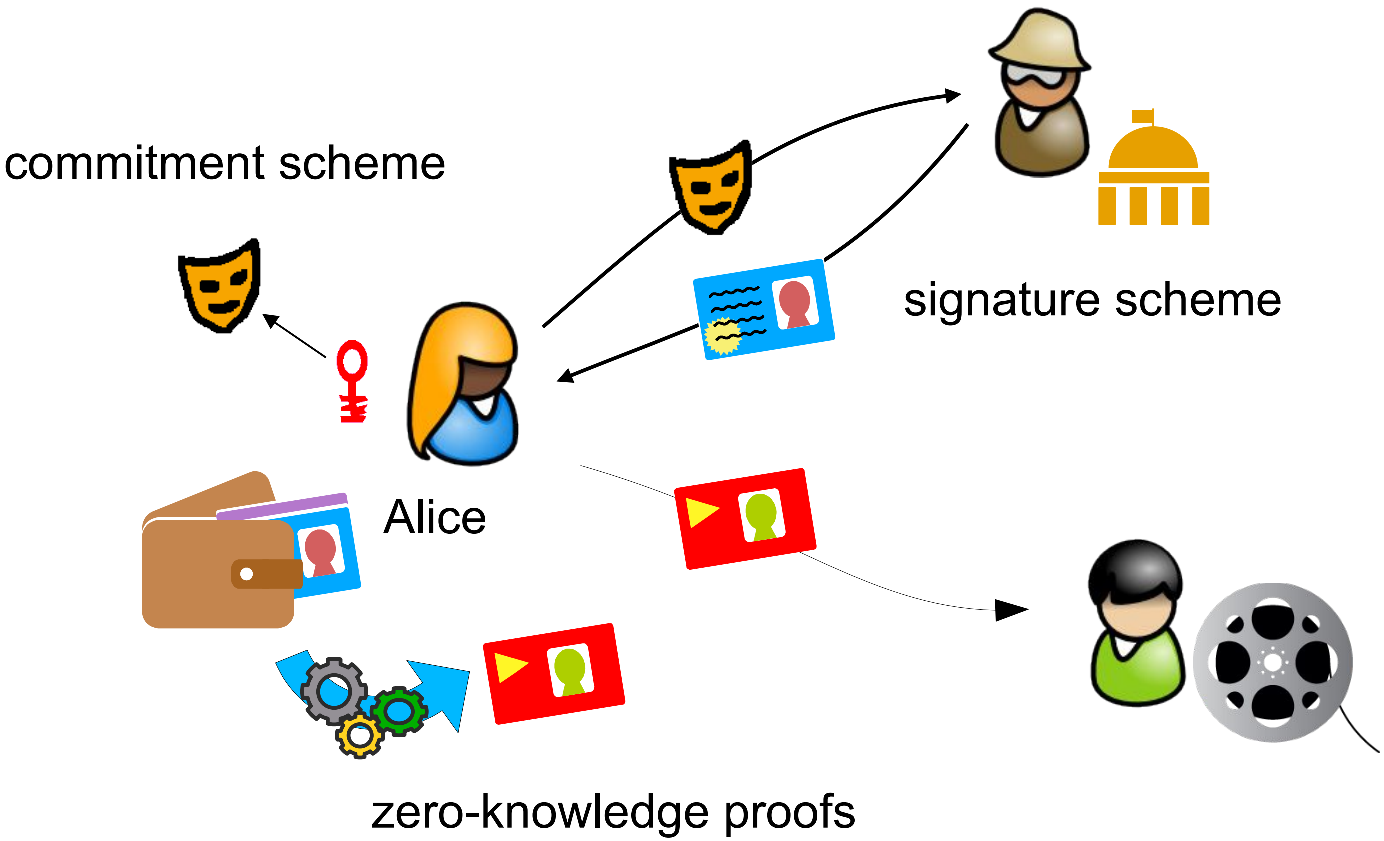


- Let $c' = c\, b^{t}$ mod n with randomly chosen t
- Then $d = c'^{e} \, a1^{m1} a2^{m2} \, b^{s-et}$ (mod n), i.e.,
  $(c', e, s^* = s-et)$ is also signature on $m1$ and $m2$

To prove knowledge of signature (c', e, s^*) on $m2$ and some $m1$

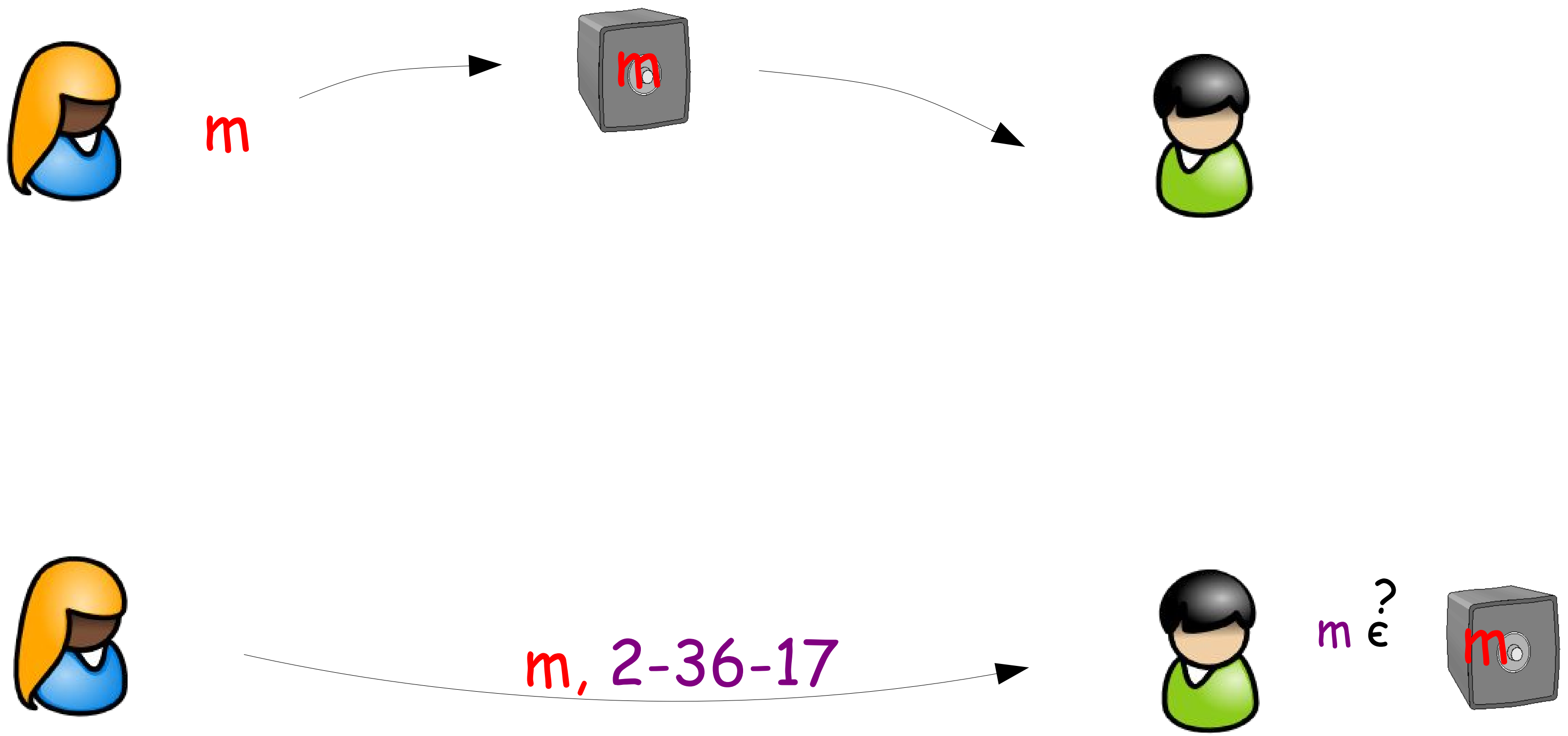
- provide c'
- $PK\{(\varepsilon, \mu1, \sigma) : \; d/a2^{m2} := c'^{\varepsilon} \, a1^{\mu1} \, b^{\sigma} \wedge \mu \in \{0,1\}^{\ell} \wedge \varepsilon > 2^{\ell+1}\}$

$\rightarrow$ proves $d := c'^{\varepsilon} \, a1^{\mu1} \, a2^{m2} b^{\sigma}$

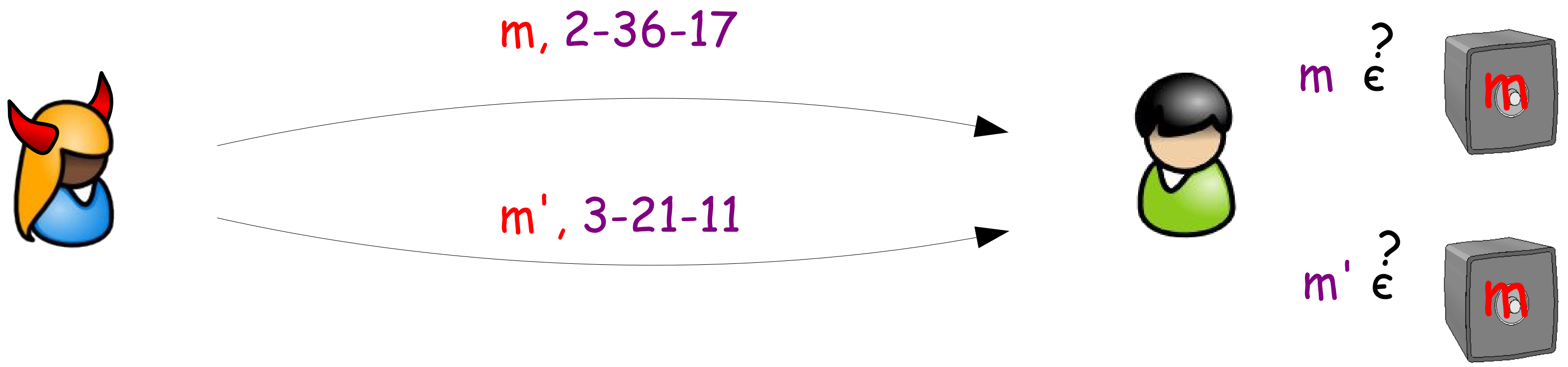

commitment scheme
signature scheme
Alice
zero-knowledge proofs

commitment scheme

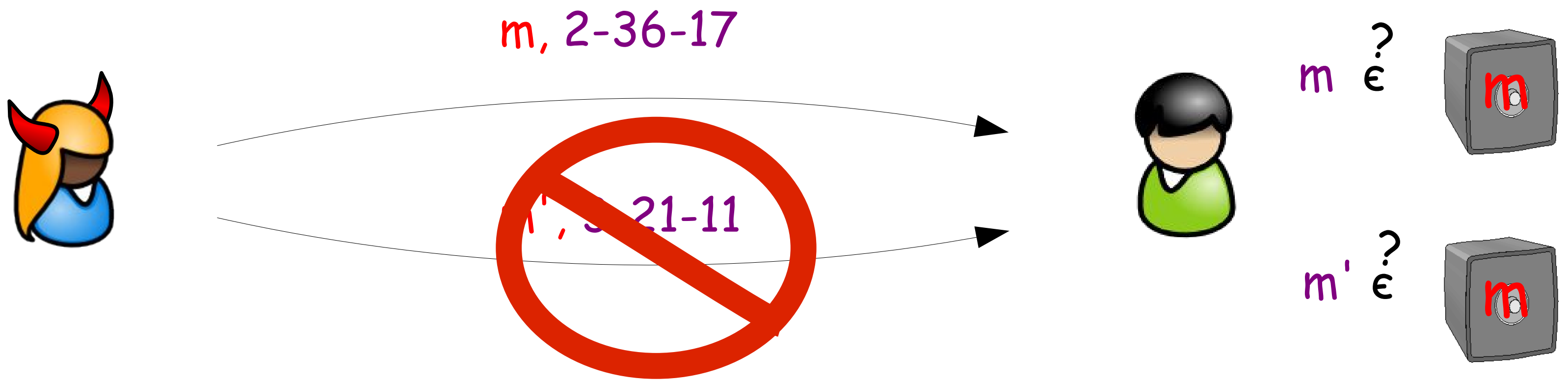
Commitment Scheme: Functionality



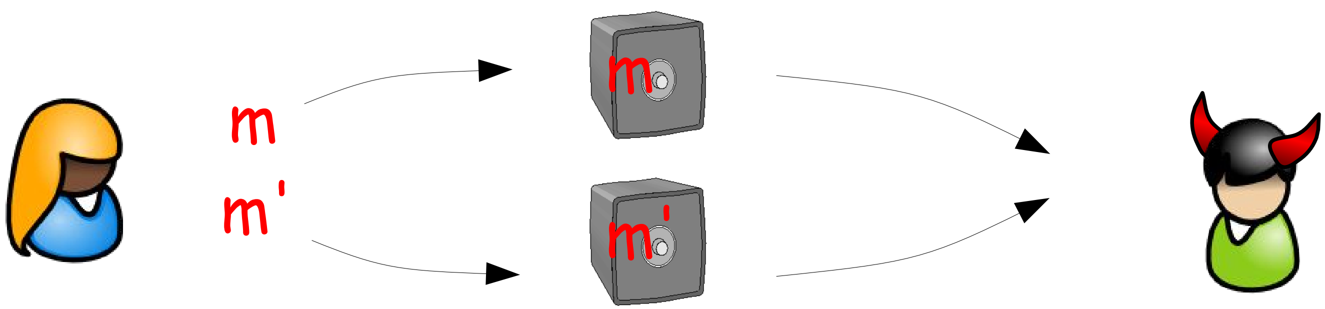
Binding



Binding

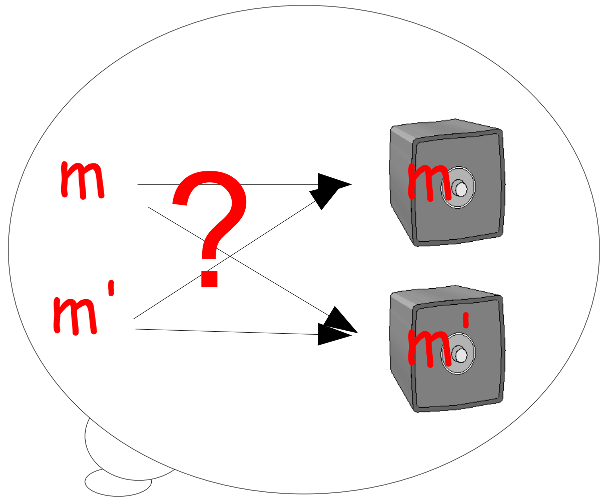


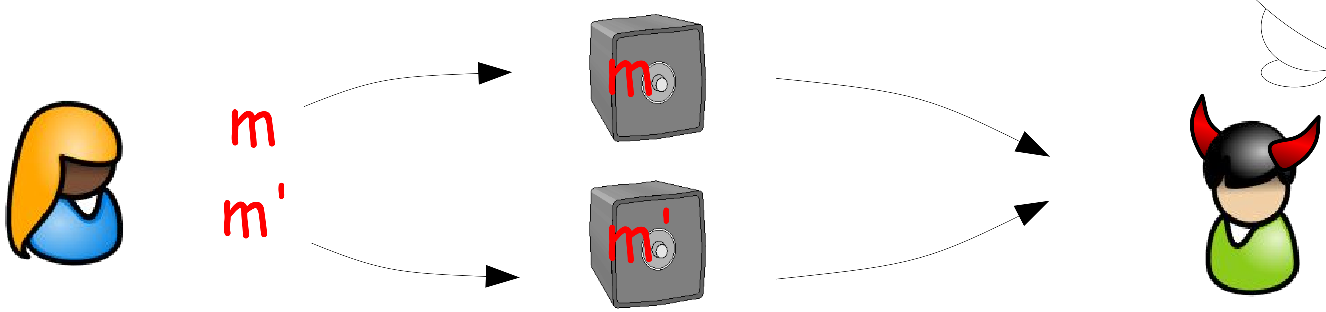
Hiding: for all message m, m'



Commitment Scheme: Security

Hiding: for all message m, m'





Group $G = \langle g \rangle = \langle h \rangle$ of order q

To commit to element $x \in Z_q$:

- Pedersen: perfectly hiding, computationally binding

  choose $r \in Z_q$ and compute $c = g^x h^r$

- ElGamal: computationally hiding, perfectly binding:

  choose $r \in Z_q$ and compute $c = (g^x h^r, g^r)$

To open commitment:

- reveal x and r to verifier
- verifier checks if $c = g^x h^r$

Pedersen's Scheme:

Choose $r \in Z_q$ and compute $c = g^x h^r$

Perfectly hiding:

Let c be a commitment and $u = \log_g h$

Thus $c = g^x h^r = g^{x+ur} = g^{(x+ur')+u(r-r')}$

$= g^{x+ur'} h^{r-r'}$ for *any* r'!

I.e., given c and x' here exist r' such that $c = g^{x'} h^{r'}$

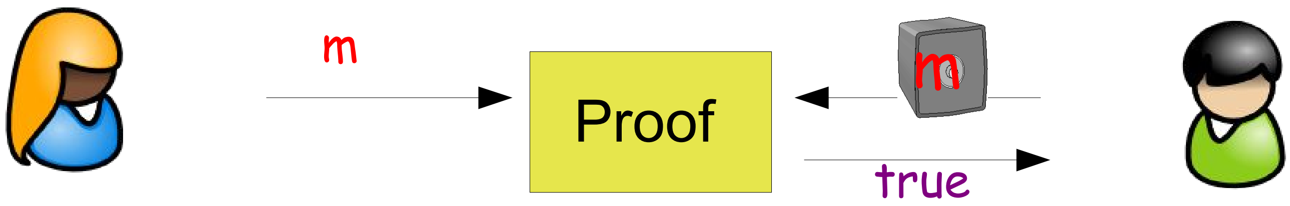
Computationally binding:

Let c, (x', r') and (x, r) s.t. $c = g^{x'} h^{r'} = g^x h^r$

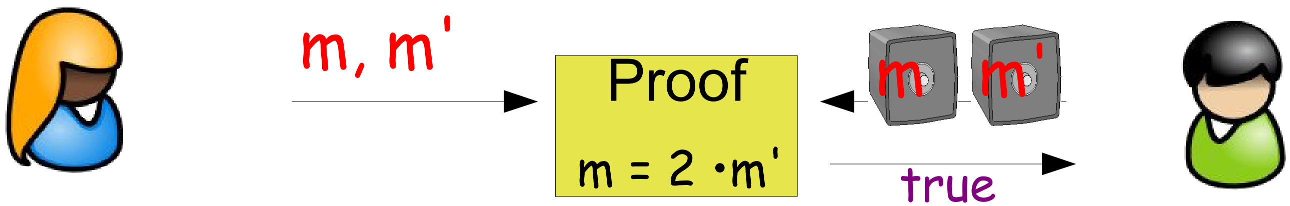
Then $g^{x'-x} = h^{r-r'}$ and $u = \log_g h = (x'-x)/(r-r') \bmod q$

Commitment Scheme: Extended Features

Proof of Knowledge of Contents



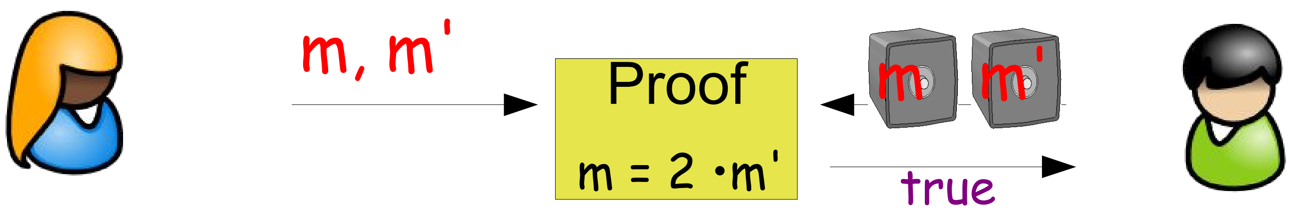
Proof of Relations among Contents



Commitment Scheme: Extended Features

Let $C1 = g^{m}h^{r}$ and $C' = g^{m'}h^{r}$ then:

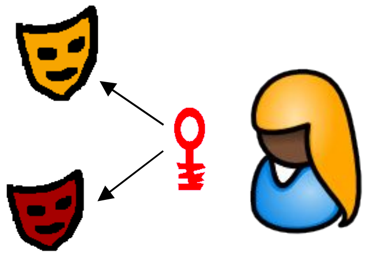
$$PK\{(\alpha,\beta): \; C = g^{\beta}h^{\alpha}\}$$



$$PK\{(\alpha,\beta,\gamma): \; C' = g^{\beta}h^{\alpha} \wedge C = (g^{2})^{\beta}h^{\gamma}\}$$


putting things together

Realizing Pseudonyms and Key Binding

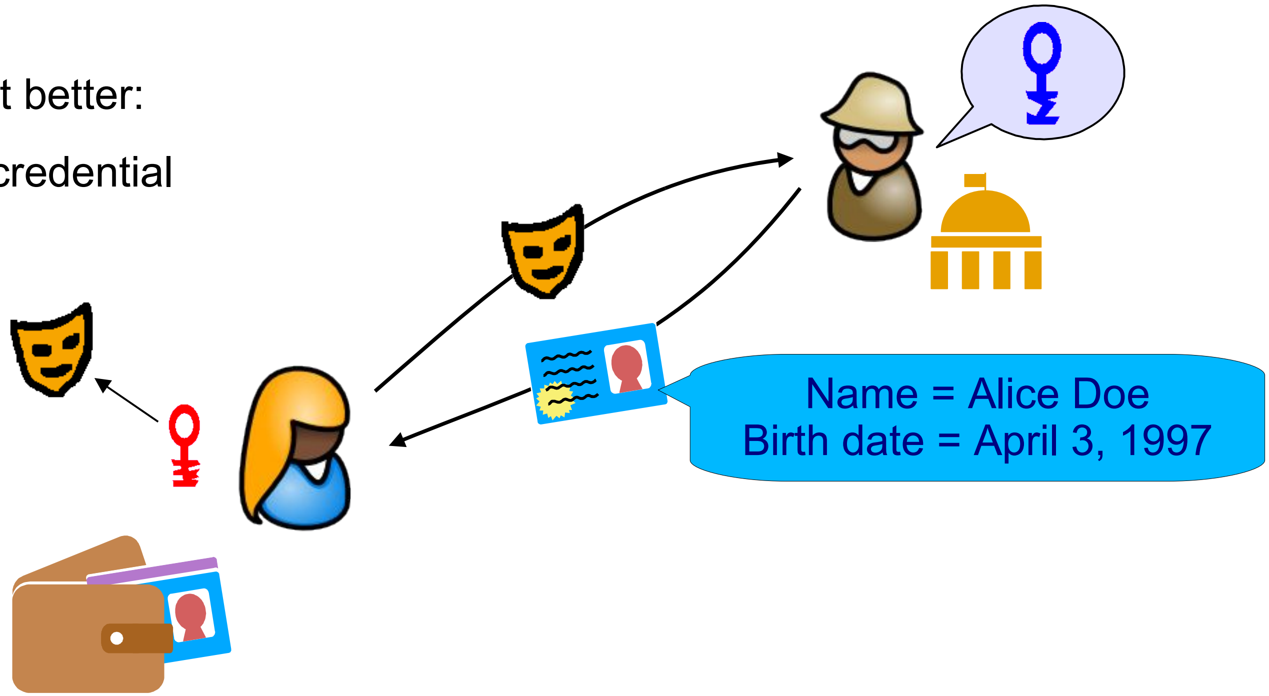


- Let $G = \langle g \rangle = \langle h \rangle$ of order q
- User's secret key: random $sk \in Zq$
- To compute a pseudonym Nym
  - Choose random $r \in Zq$
  - Compute $Nym = g^{sk}h^{r}$

Privacy-protecting authentication with Privacy ABCs

Like PKI, but better:

- Issuing a credential

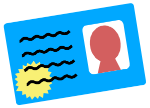


Concept: credentials

Realizing Issuance of Credential

Recall: a signature (c,e,s) on messages $m1, \dots, mk$:

- $m1, \dots, mk \in \{0,1\}^{\ell}$:
- $e > 2^{\ell+1}$
- $d = c^{e} a_1^{m1} \cdot \ldots \cdot a_k^{mk} b^{s}$ mod n

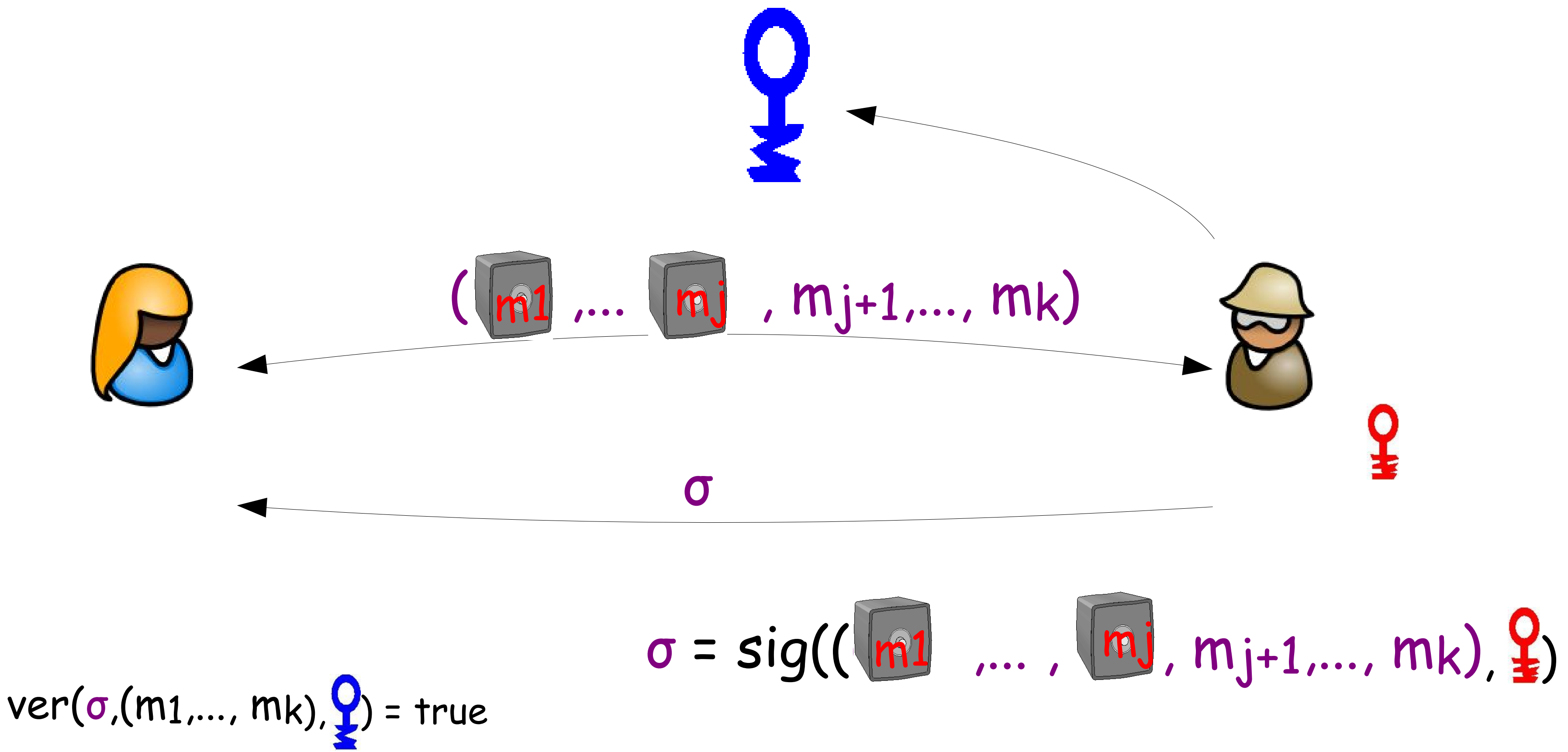


Problem: Pseudonym not in message space!

Solution: Sign secret key instead

$\rightarrow$ $d = c^{e} a_1^{sk} \cdot a_2^{m2} \cdot \ldots \cdot a_k^{mk} b^{s}$ mod n

New Problem: how can we sign a secret message?



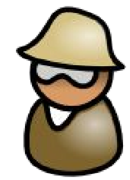
Verification remains unchanged!

Security requirements basically the same as for signatures, but

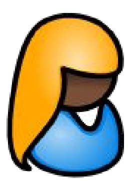
- signer should not learn any information about m1, ..., mj
- Forgery w.r.t. message clear parts and opening of commitments

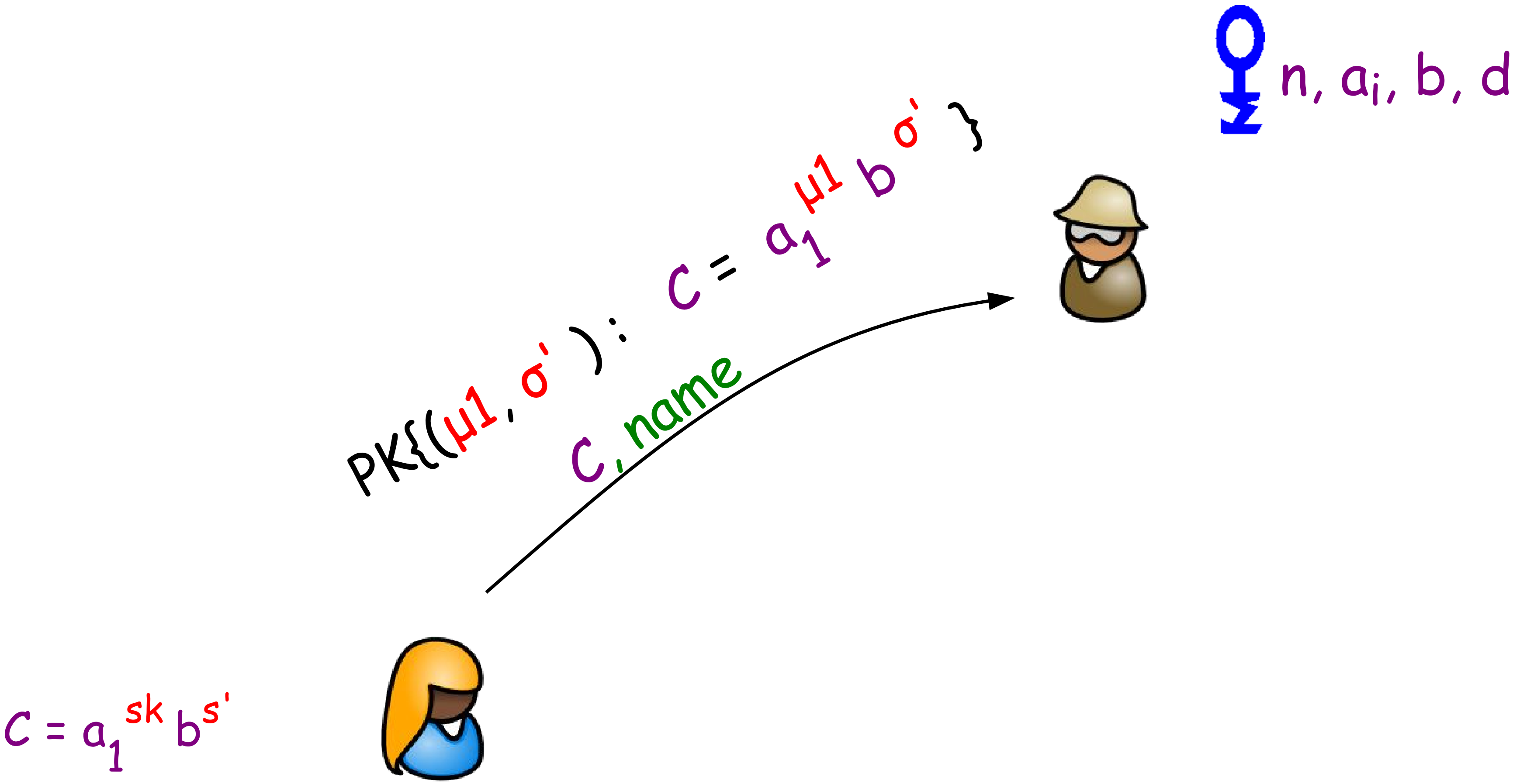
Realizing Issuance of Credential

n, a_i, b, d



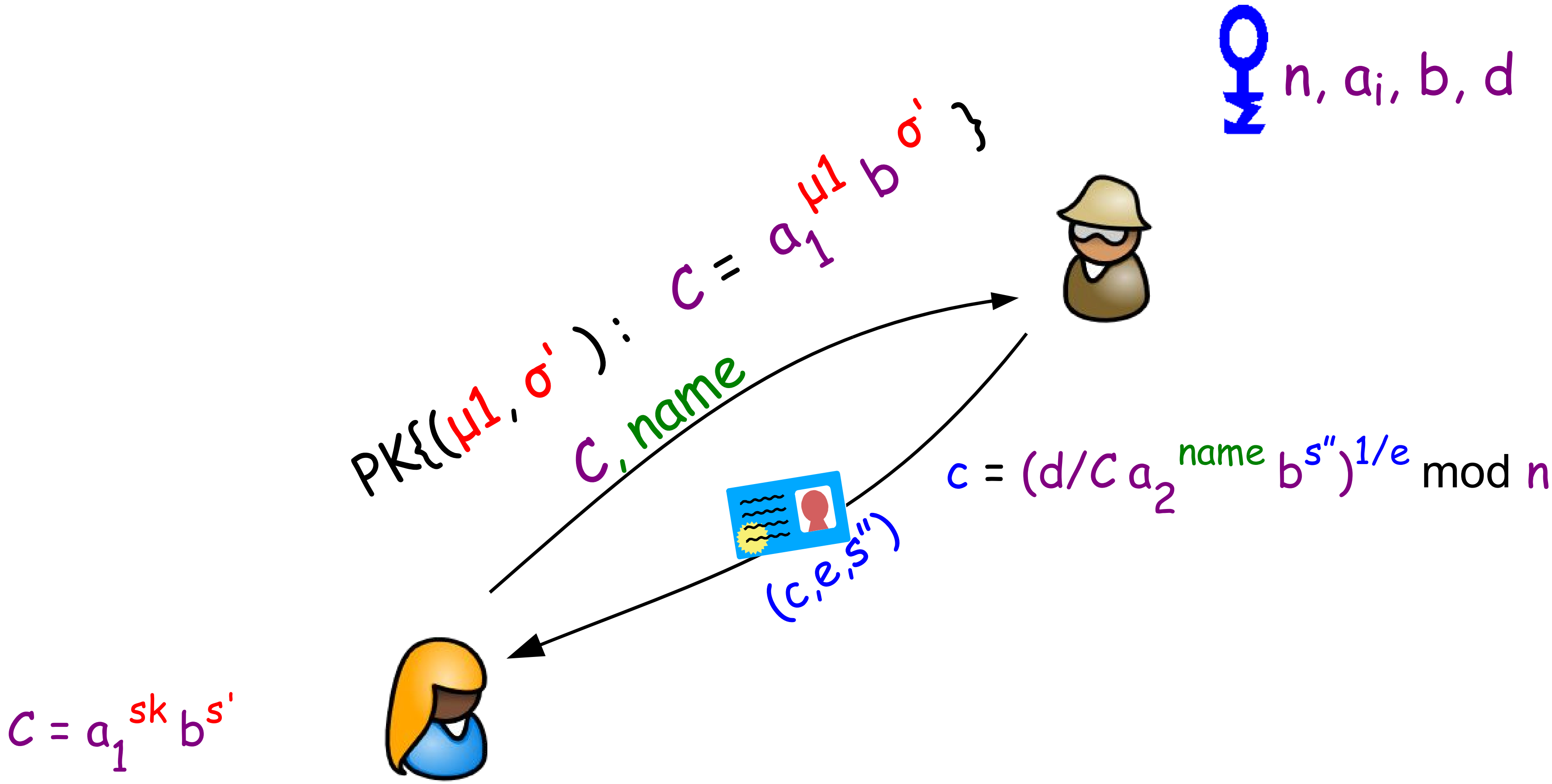
$C = a_1^{sk} b^{s'}$



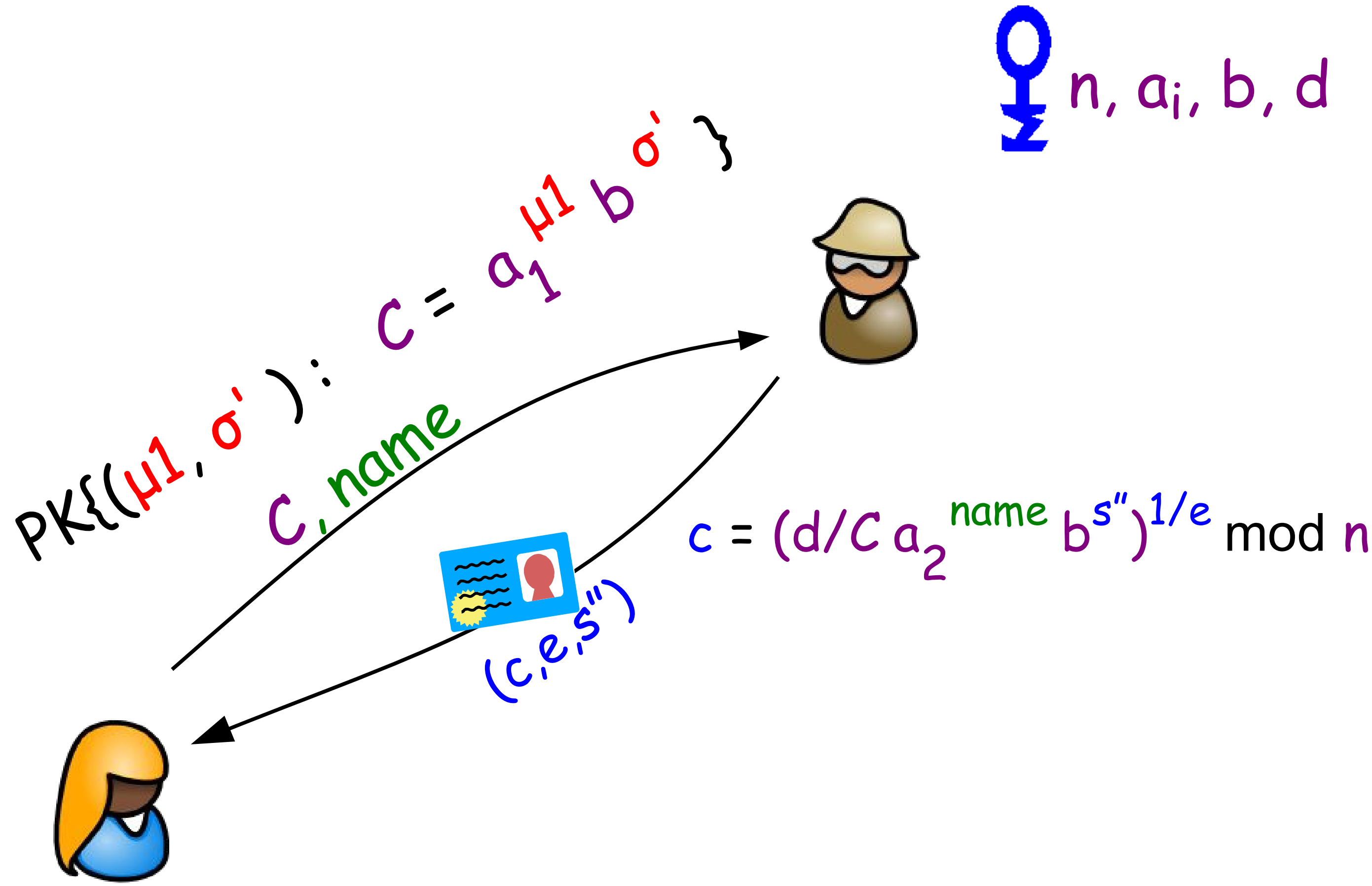


n, a_i, b, d
PK{(μ1, σ'): C = a_1^μ1 b^σ' }
C, name
C = a_1^sk b^s'

Realizing Issuance of Credential

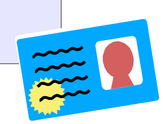


Realizing Issuance of Credential

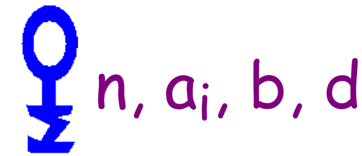


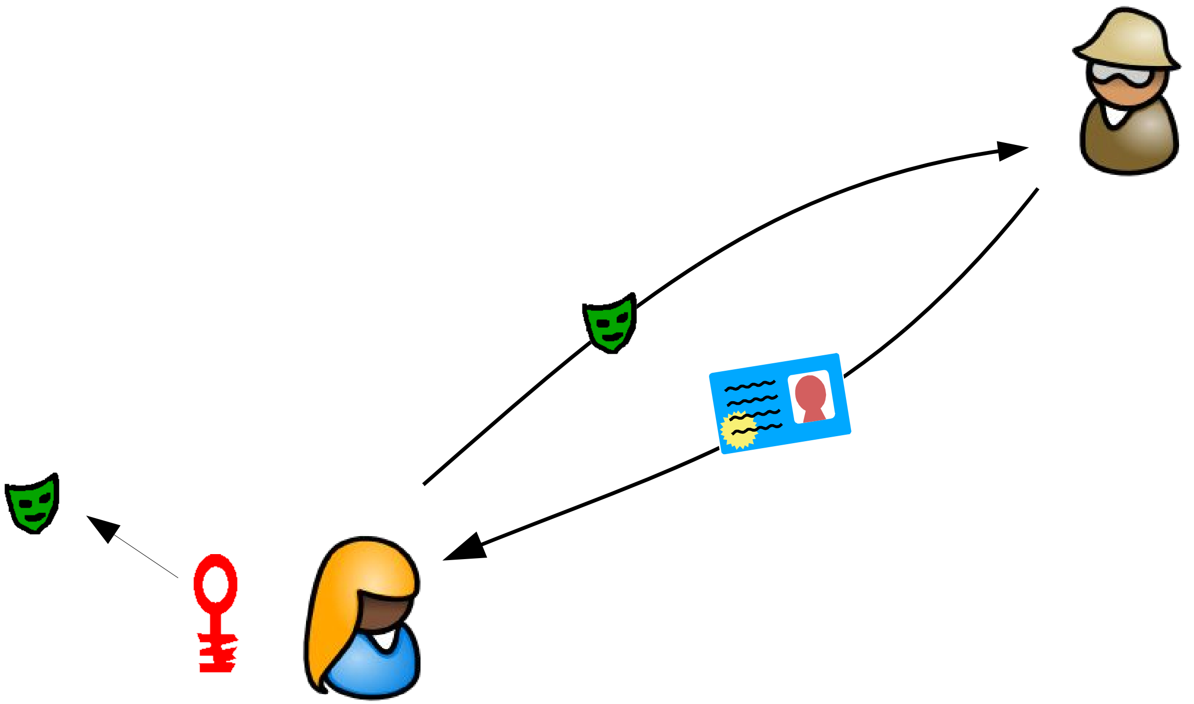
$C = a_1^{sk} b^{s'}$

$$d = c^e a_1^{sk} a_2^{name} b^{s'' + s'} \pmod n$$



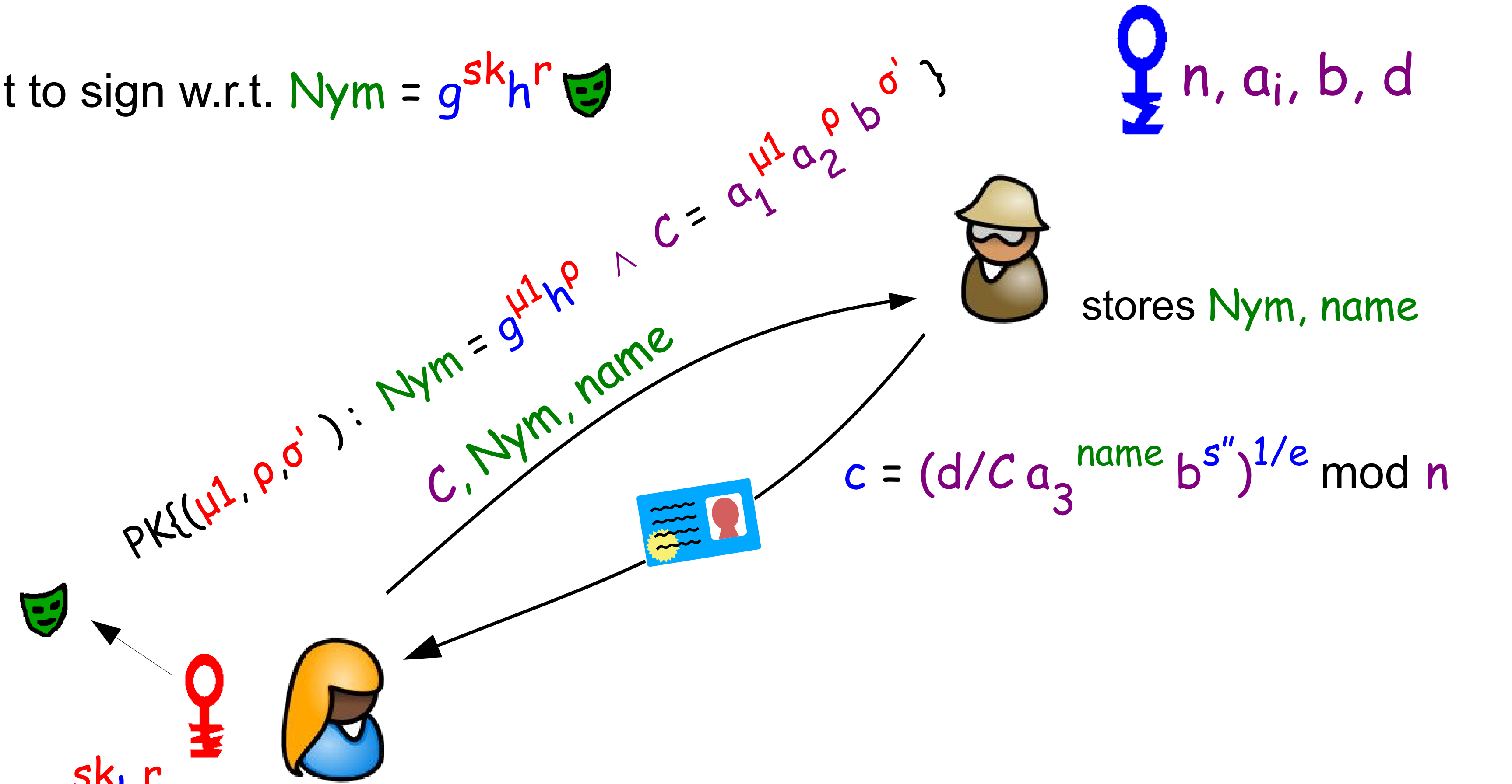
Want to sign w.r.t. Nym = $g^{sk}h^{r}$





IBM

Want to sign w.r.t. Nym $= g^{sk} h^{r}$



Nym $= g^{sk} h^{r}$

$C = a_1^{sk} a_2^{r} b^{s'}$

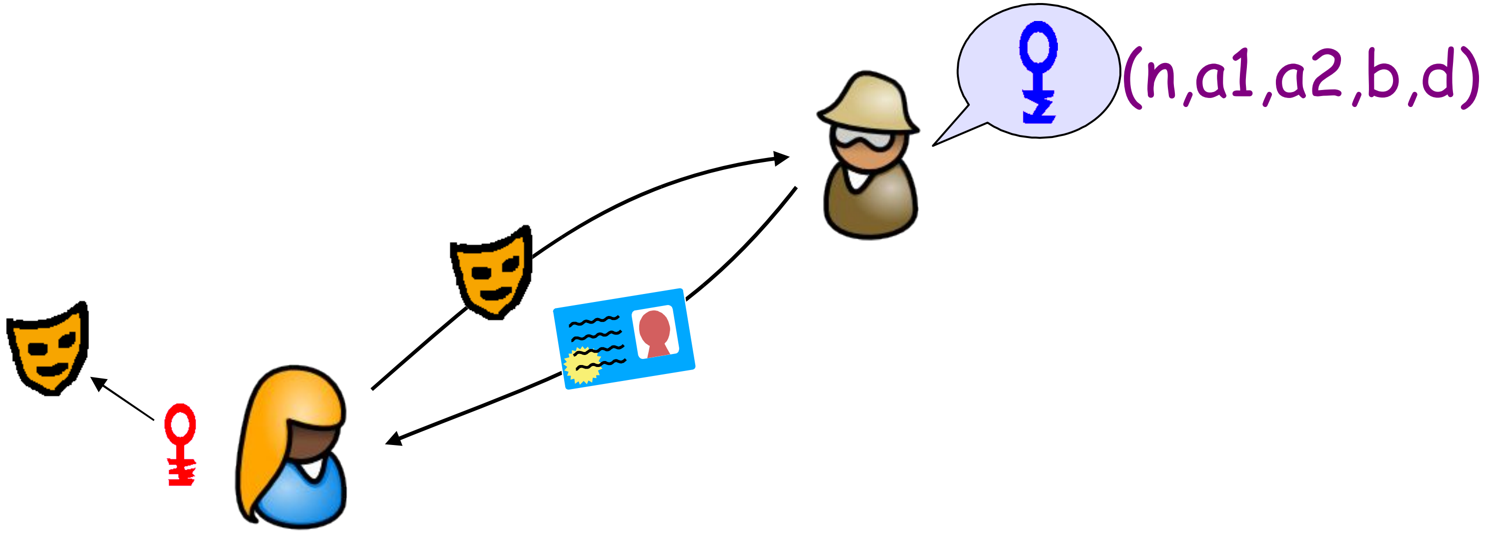
$$d = c^{e} a_1^{sk} a_2^{r} a_3^{name} b^{s'' + s'} \pmod{n}$$

An Example Scenario

Polling: Scenario and Requirements

Scenario:

- Pollster(s) and a number of users
- Only registered user (e.g., students who took a course) can voice opinion (e.g., course evaluation)
- User can voice opinion only once (subsequent attempts are dropped)
- Users want to be anonymous
- A user's opinion in different polls must not be linkable

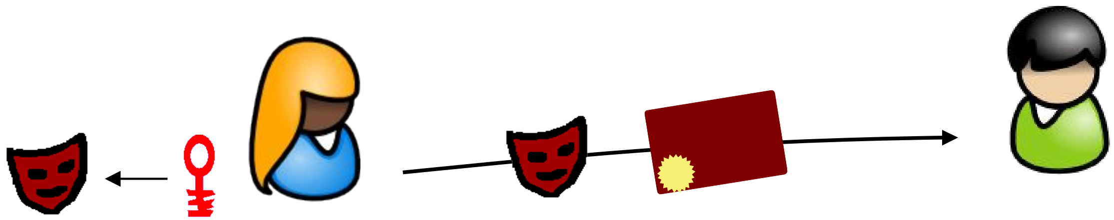


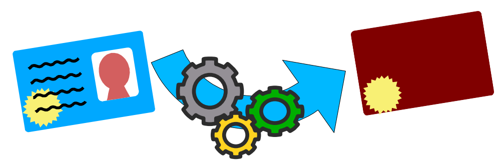
- User generates pseudonym (ID for registration)
- User obtains credential on pseudonym stating that she is eligible for polls, i.e., (c,e,s)

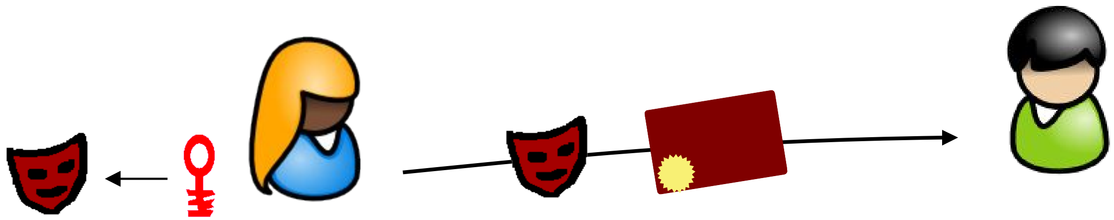
$$d = c^{e} a_1^{sk} a_2^{r} a_3^{attr} b^{s} \pmod{n}$$

- Credential can contain attributes (e.g., course ID) about her

Polling – Solution: Submit Poll



1. User generates domain pseudonym, domain = pollID 
2. User transforms credential 
3. Transformed credential with a subset of the attributes
   – User is anonymous and unlinkable
   – Multiple opinions are detected because uniqueness of domain pseudonym

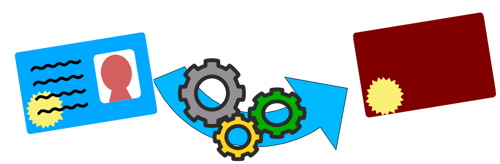


1. Domain pseudonym: $P = g_d^{sk} = H(\mathrm{pollID})^{sk}$

$P1 = H(\mathrm{pollID1})^{sk}$ and $P2 = H(\mathrm{pollID2})^{sk}$ are unlinakble

(under the Decisional Diffie-Hellman assumption)

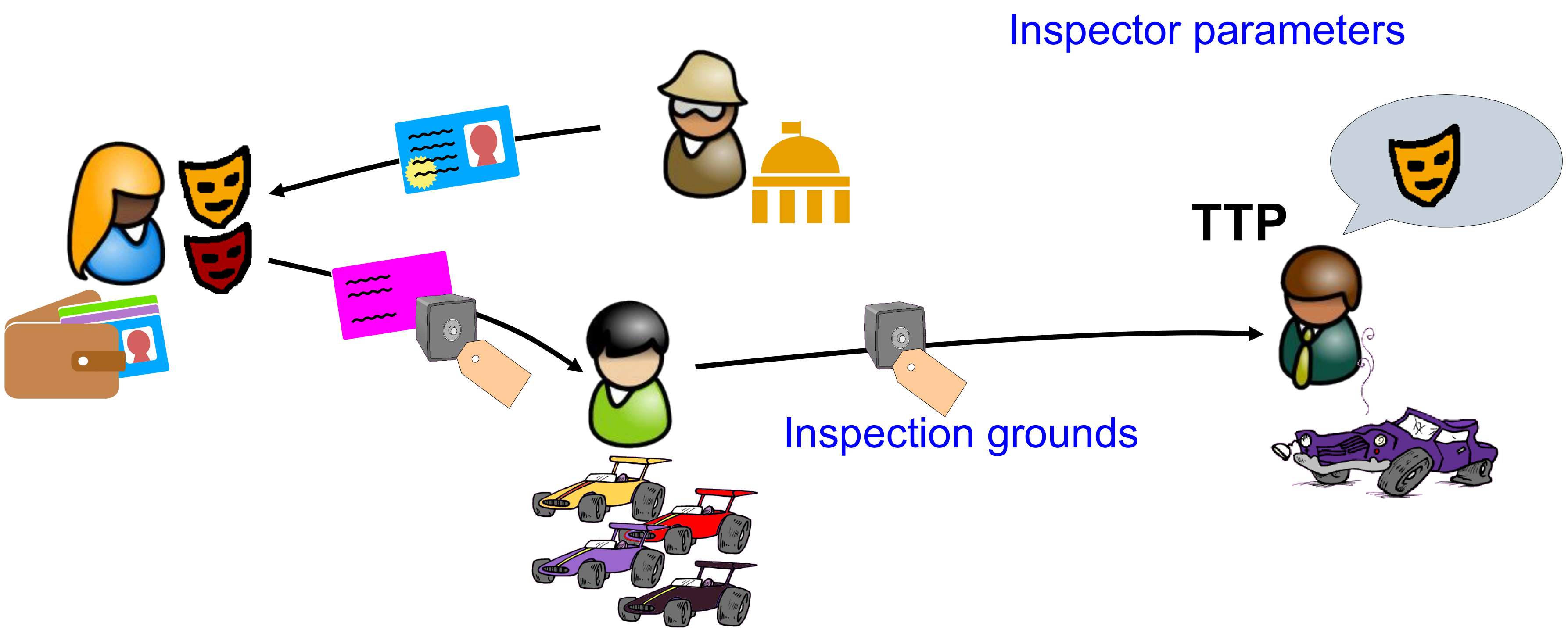
2. User transforms credential:



- $c' = c\, b^{s'}$ mod n with randomly chosen s'
- $SPK\{(\varepsilon, \mu 1, \mu 2, \mu 3, \sigma) : P = g_d^{\mu 1} \wedge d := c'^{\varepsilon} a1^{\mu 1} a2^{\mu 2} a3^{\mu 3} b^{\sigma} \pmod{n} \wedge \mu 1, \mu 2, \mu 3 \in \{0,1\}^{\ell} \wedge \varepsilon > 2^{\ell+1}\}(\mathrm{opinion})$

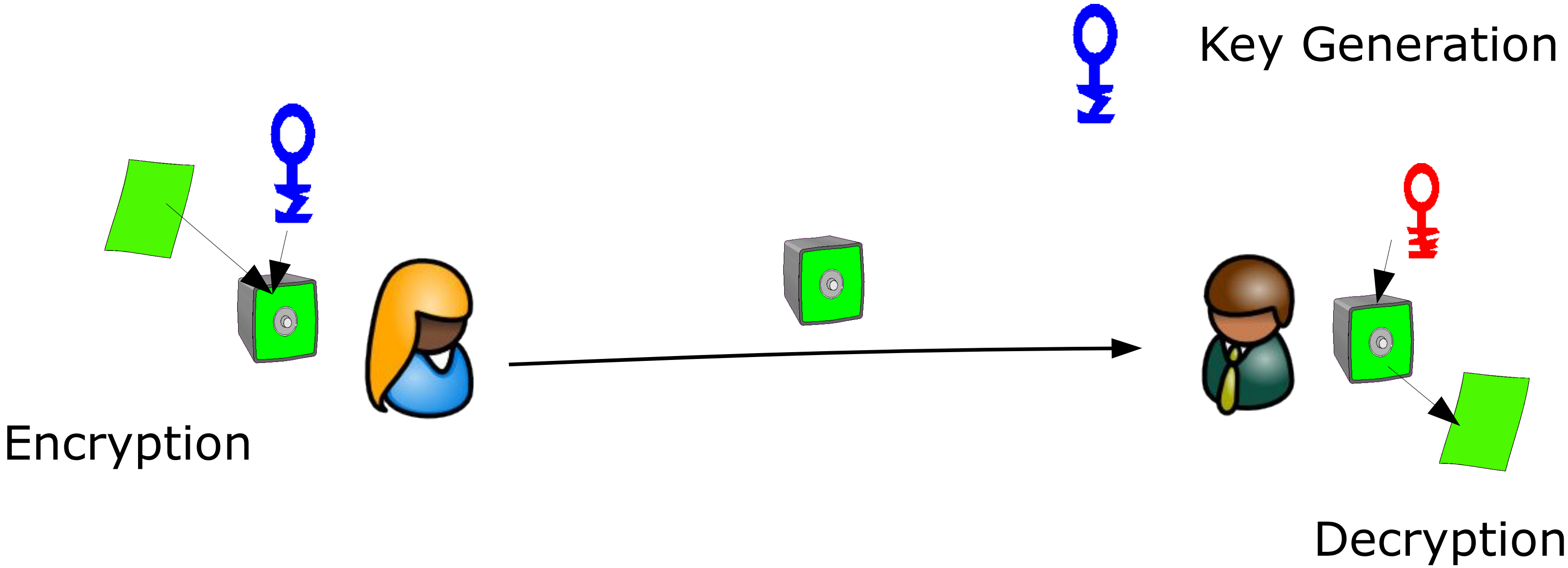
Further Concepts

Concept – Inspection

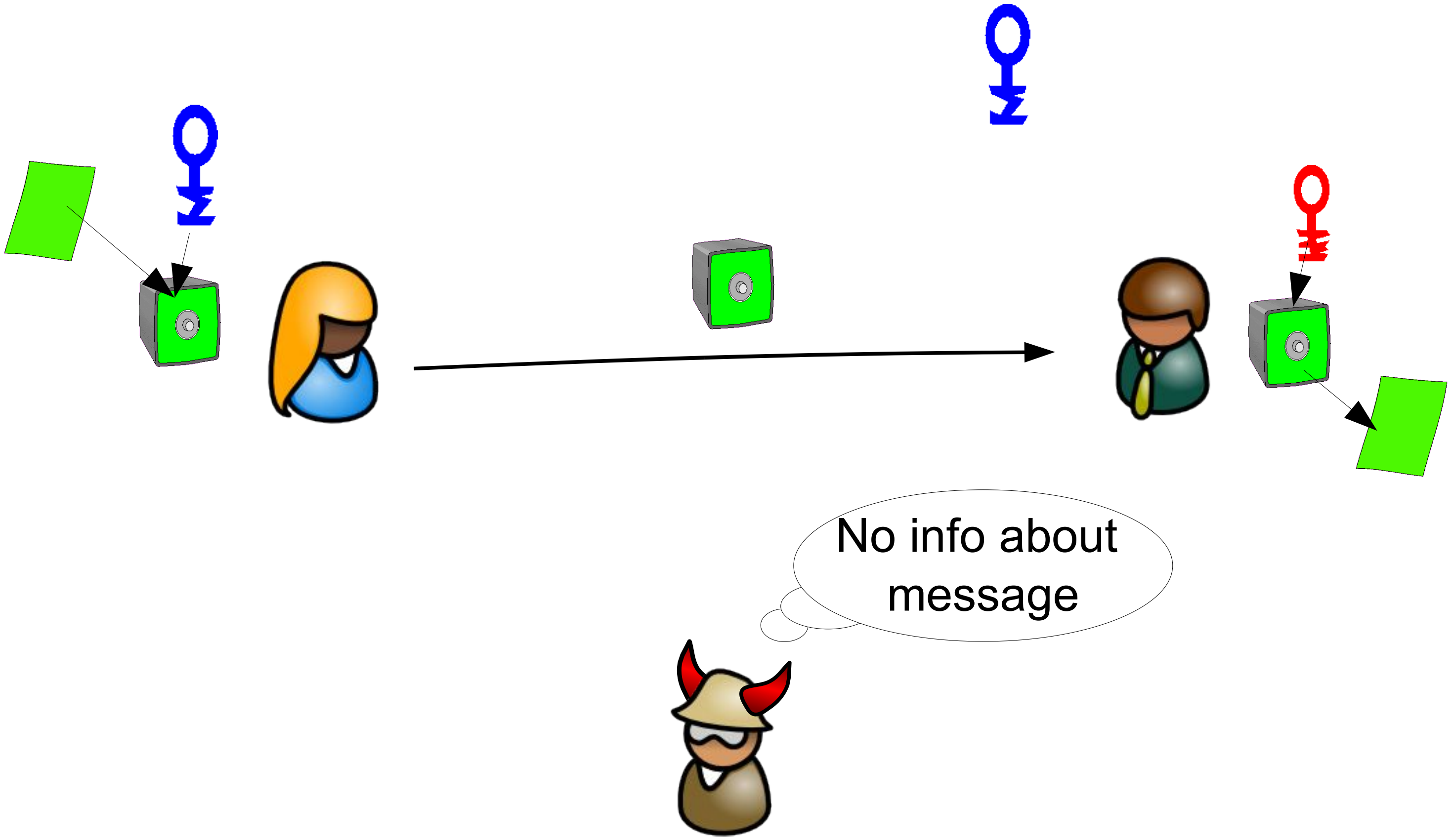


- If car is damaged: ID with insurance or gov't needs be retrieved
- Similarly: verifiably encrypt any certified attribute *(optional)*
- TTP is off-line & can be distributed to lessen trust

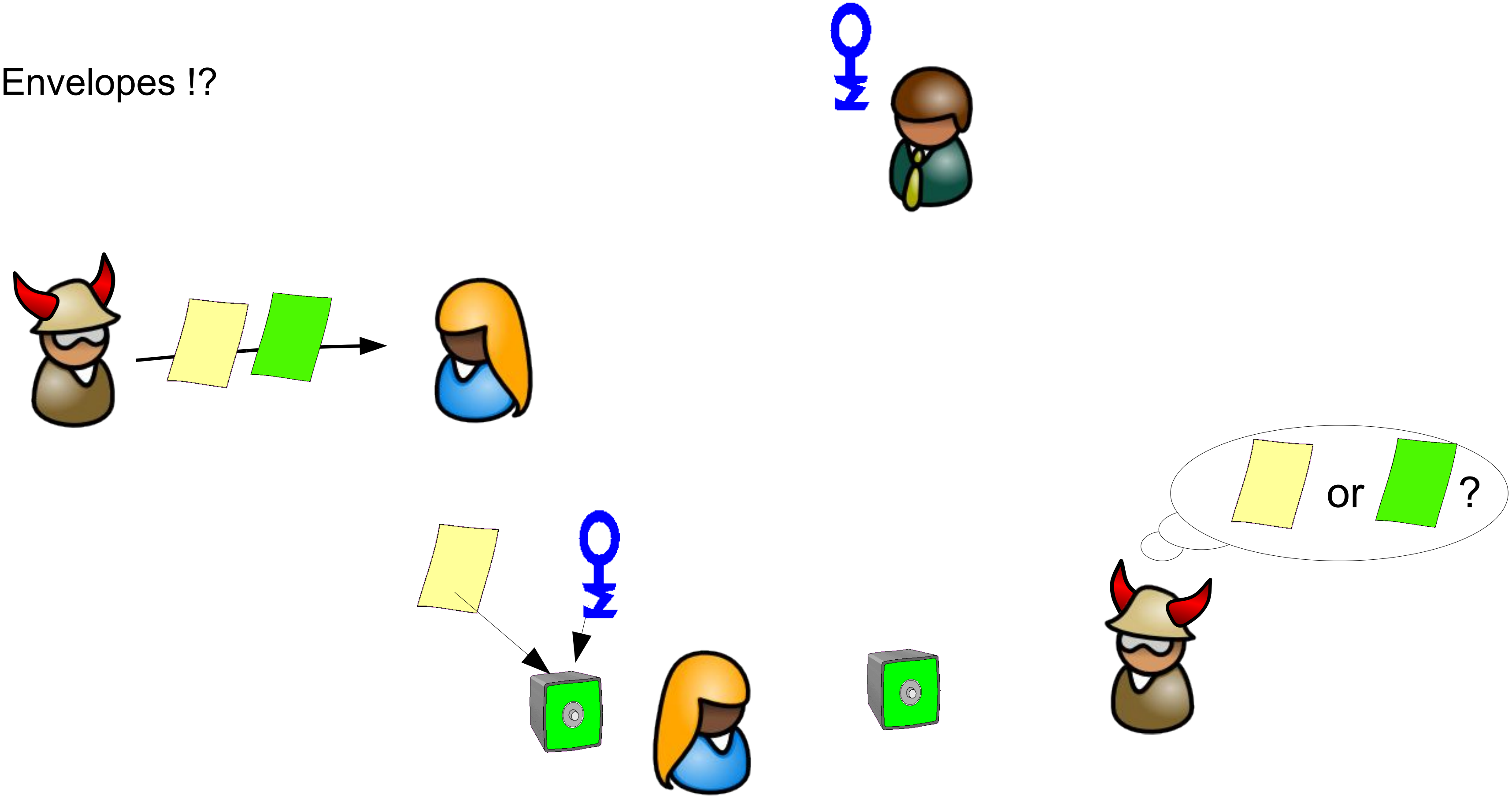
Public Key Encryption



Like Envelopes !?

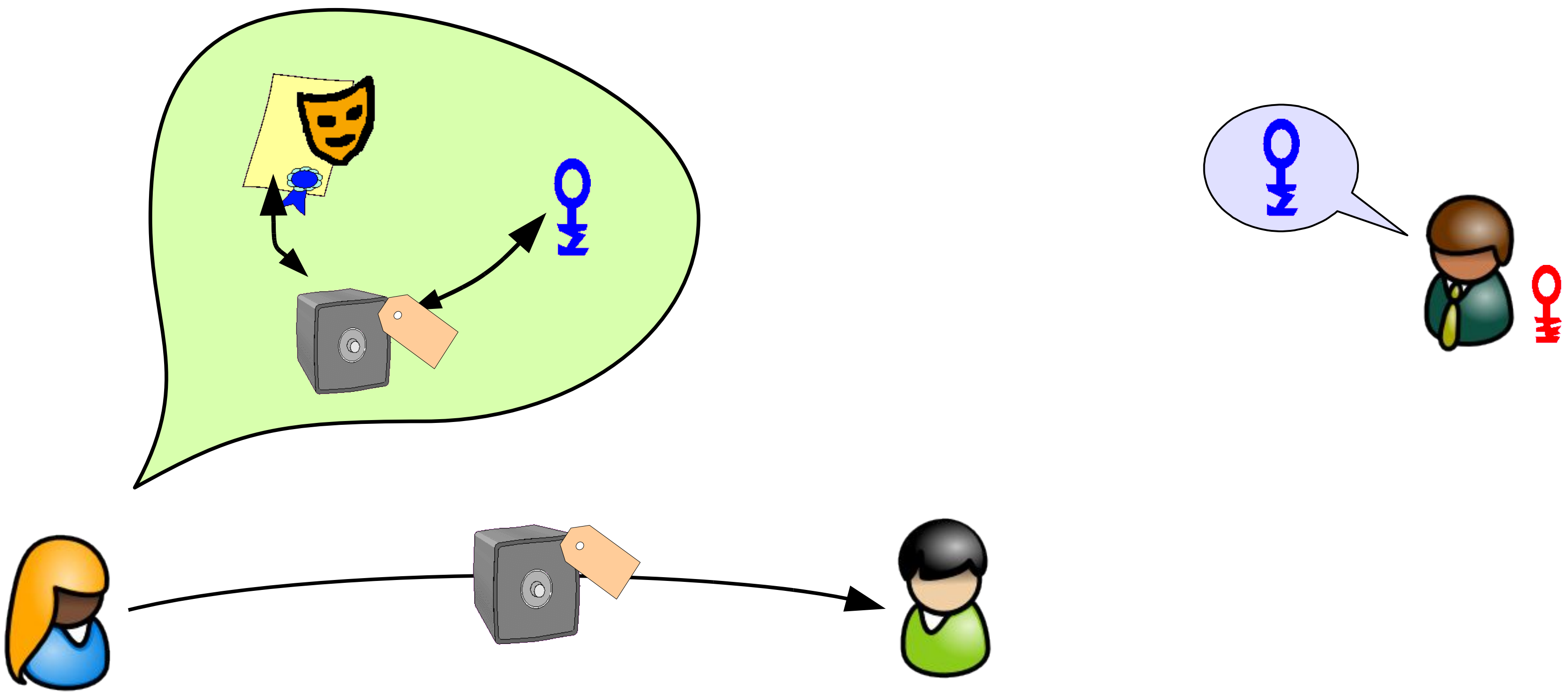


Like Envelopes !?



This is called *semantic security* (secure if used once only or within careful construction.)

Verifiable Encryption with Label

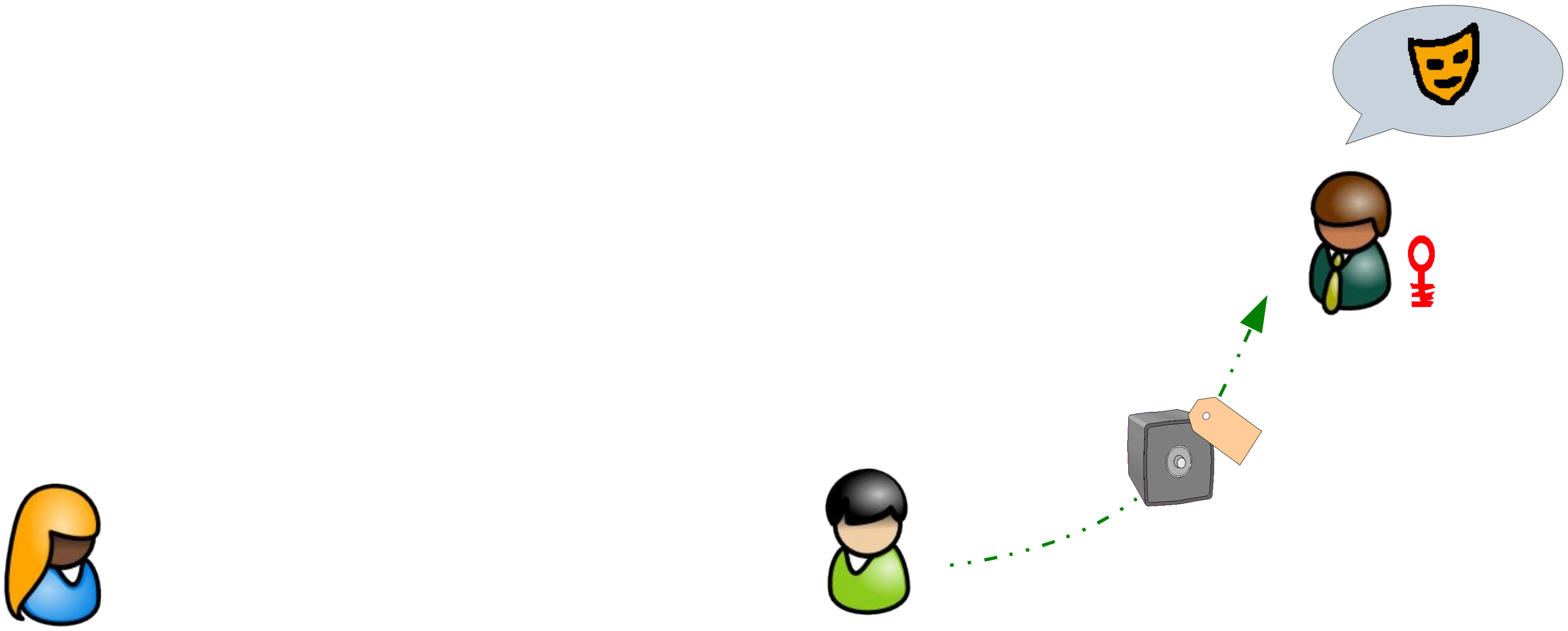


Label is important to bind context to an encryption.

E.g., defines decryption condition, binds user to car, etc.

Security definition: change of label is new ciphertext

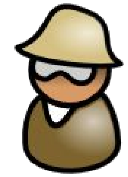
Verifiable Encryption with Label



- Of attributes (discrete logarithm)
  – Camenisch-Shoup (SRSA) – based on Paillier Encryption
- Of pseudonyms (group elements)
  – Cramer-Shoup (DL) or rarely ElGamal (DL)
- Otherwise (any secret for which ZKPK exists)
  – Camenisch-Damgaard, works for any scheme, but much less efficient
- ....Open Problem to find new ones!

ElGamal Encryption Scheme

- Group $G = \langle g \rangle$ of order q
- Secret Key Group $x \in \{1,\ldots,q\}$; Public key $y = g^x$
- To encrypt message $m \in \langle g \rangle$:
  - choose random $r \in \{1,\ldots,q\}$;
  - compute $c = (y^r m, g^r)$
- To decrypt ciphertext $c = (c_1,c_2)$
  - We know $c = (y^r m, g^r) = (g^{xr} m, g^r)$
  - Thus set $m = c_1 c_2^{-x} = y^r m g^{-xr} = y^{r-r} m = m$



Nym

$Nym = g^{sk}h^{r}$

$d = c^{e} a_1^{sk} a_2^{r} a_3^{name} b^{s'' + s'} \pmod{n}$

$y = g^{x}$

- Encrypt Nym : random $u \in \{1,\dots,q\}$ and $enc = (y^{u}\, Nym,\, g^{u}) = (e1,e2)$
- Compute proof token (presentation token):
  - compute $c' = c\, b^{t} \bmod n$ with randomly chosen t
  - compute proof

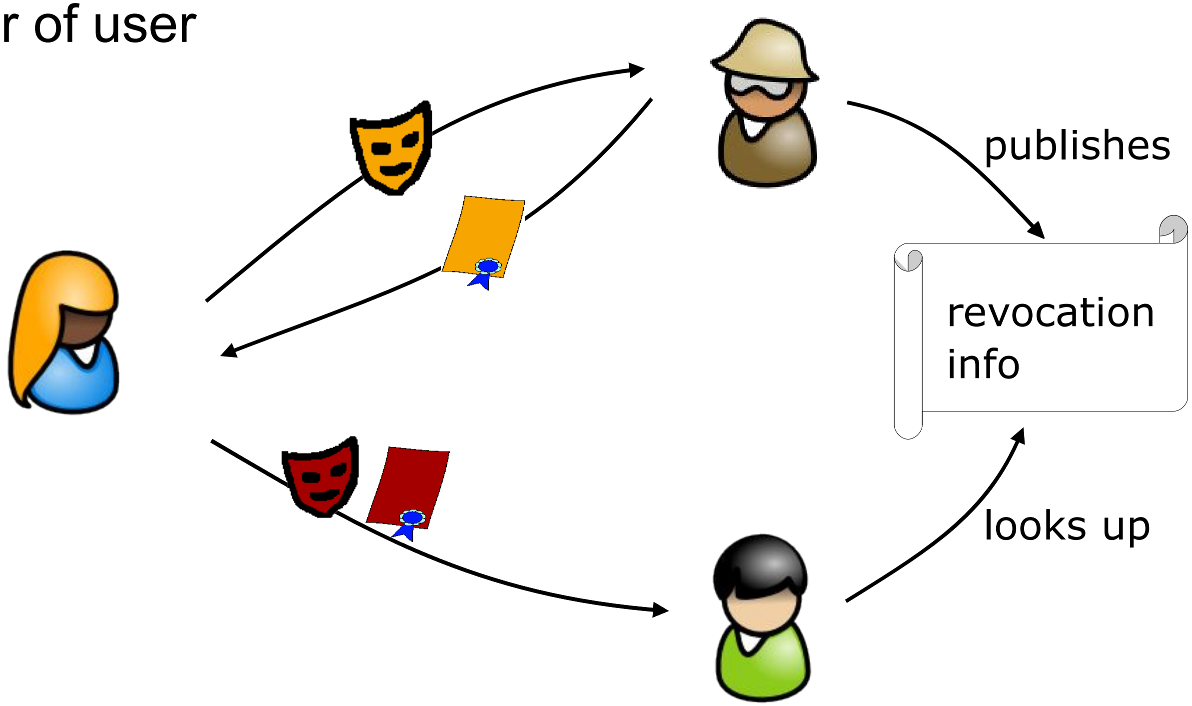
$$PK\{(\varepsilon, \mu1, \mu2, \mu3, \sigma) : d := c'^{\varepsilon}\, a1^{\mu1} a2^{\mu2} a3^{\mu3} b^{\sigma} \wedge e1 = y^{\rho}\, g^{\mu1} h^{\mu2} \wedge e2 = g^{\rho} \wedge \mu1, \mu3, \mu3 \in \{0,1\}^{\ell} \wedge \varepsilon > 2^{\ell+1} \}$$

Revocation of credentials

Anonymous Credential Revocation

various reasons to revoke credential

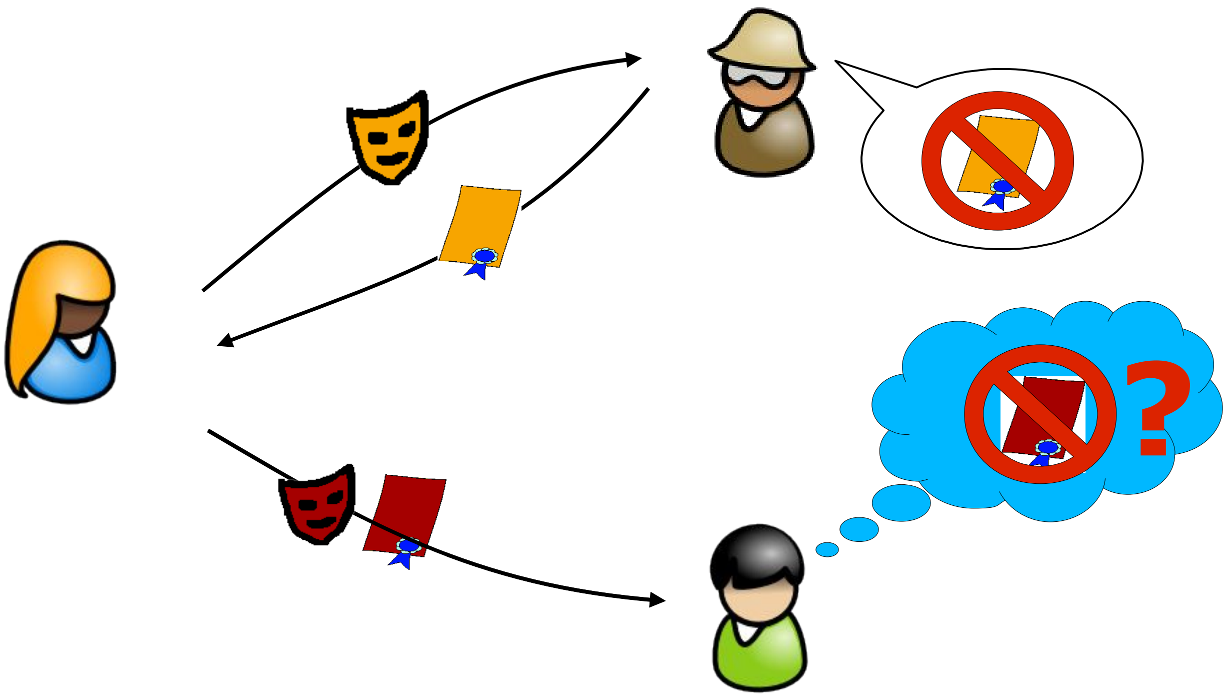
- user lost credential / secret key
- misbehavior of user



Alice should be able to convince verifier that her credential is among the good ones!

Anonymous Credential Revocation

- Pseudonyms ⟶ standard revocation lists don't work



Revocable Credentials: First Solution

- Include into credential some credential ID ui as message, e.g.,

$$d = c^{e} a_1^{sk} a_2^{ui} b^{s'' + s'} \pmod{n}$$

- Publish list of all valid (or invalid) ui's.

$$(u1, \ldots, uk)$$

- Alice proves that her ui is on the list.
  - Choose random g
  - Compute $Uj = g^{uj}$ for uj in $(u1, \ldots, uk)$
  - Prove $PK\{(\varepsilon, \mu, \rho, \sigma) : (d = c'^{\varepsilon} a_1^{\rho} a_2^{\mu} b^{\sigma} (n) \wedge U1 = g^{\mu}) \vee \cdots \vee (d = c'^{\varepsilon} a_1^{\rho} a_2^{\mu} b^{\sigma} \pmod{n} \wedge Uk = g^{\mu}) \}$

- Not very efficient, i.e., linear in size k of list :-(

Revocable Credentials: Second Solution

- Include into credential some credential ID ui as message, e.g.,

$$d = c^{e} a_1^{sk} a_2^{ui} b^{s'' + s'} \pmod{n}$$

- Publish list of all *invalid* ui's.

(u1,..., uk)

- Alice proves that her ui is not on the list.
  - Choose random h and compute $U = h^{ui}$
  - Prove $PK\{(\varepsilon, \mu, \rho, \sigma) : d = c'^{\varepsilon} a_1^{\rho} a_2^{\mu} b^{\sigma} \pmod{n}$

    $\wedge \; U = h^{\mu} \}$
  - Verifier checks whether $U = h^{uj}$ for all uj on the list.
- Better, as *only verifier* needs to do linear work (and it can be improved using so-call batch-verification...)
- What happens if we make the list of all valid ui's public?
- If credential is revoked, all past transactions become linkable...

Variation: verifier could choose h and keep it fixed for a while

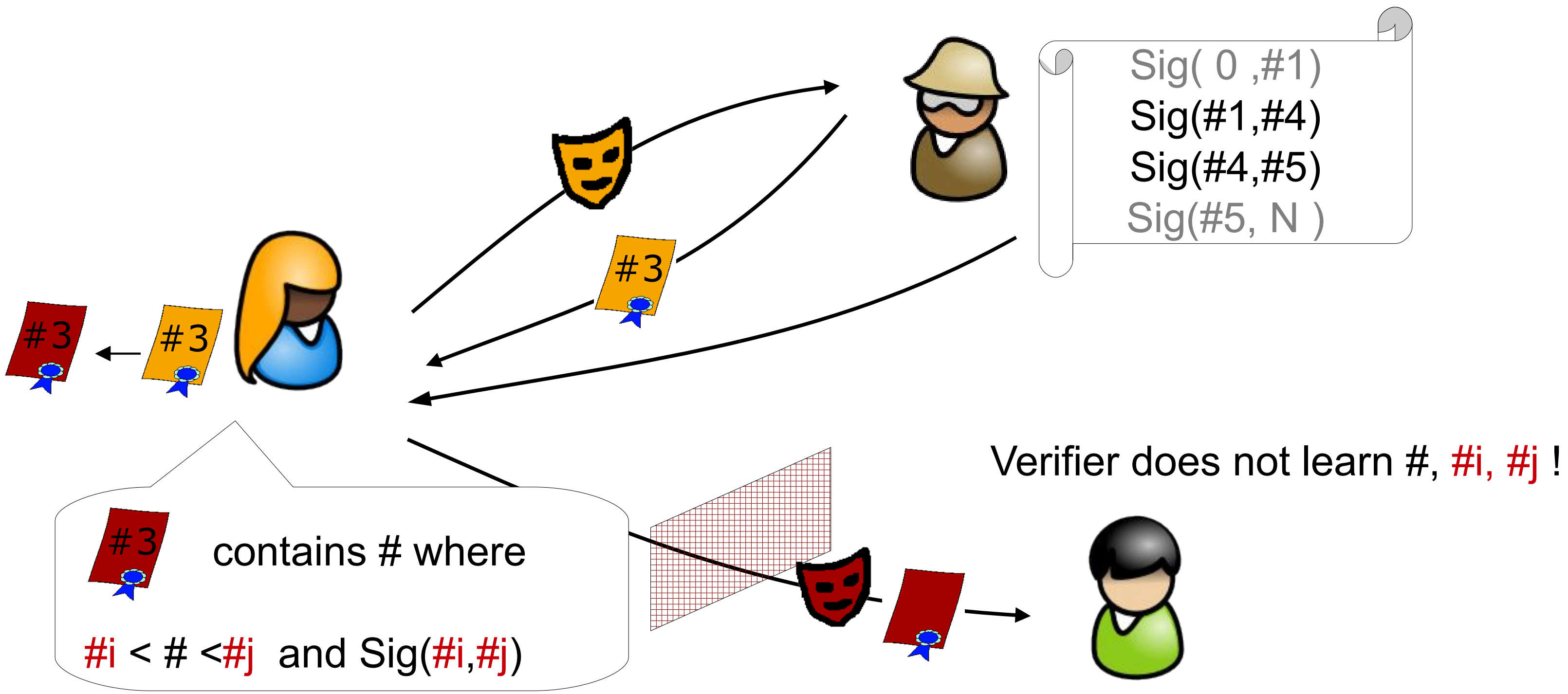
- Can pre-compute list $U_i = h^{u_i}$
- → single table lookup
- BUT: if user comes again, verifier can link!!!
- ALSO: verifier could not change h at all! or use the same as other verifiers!
  - one way out h = H(verifier, date), so user can check correctness.
  - date could be the time up to seconds and the verifier could just store all the lists, i.e., pre-compute it.

Revocable Credentials: Second Solution

... better implementation of proof :

Issuer signs intervals between revoked #

→ revocation list: #1,#4,#5,



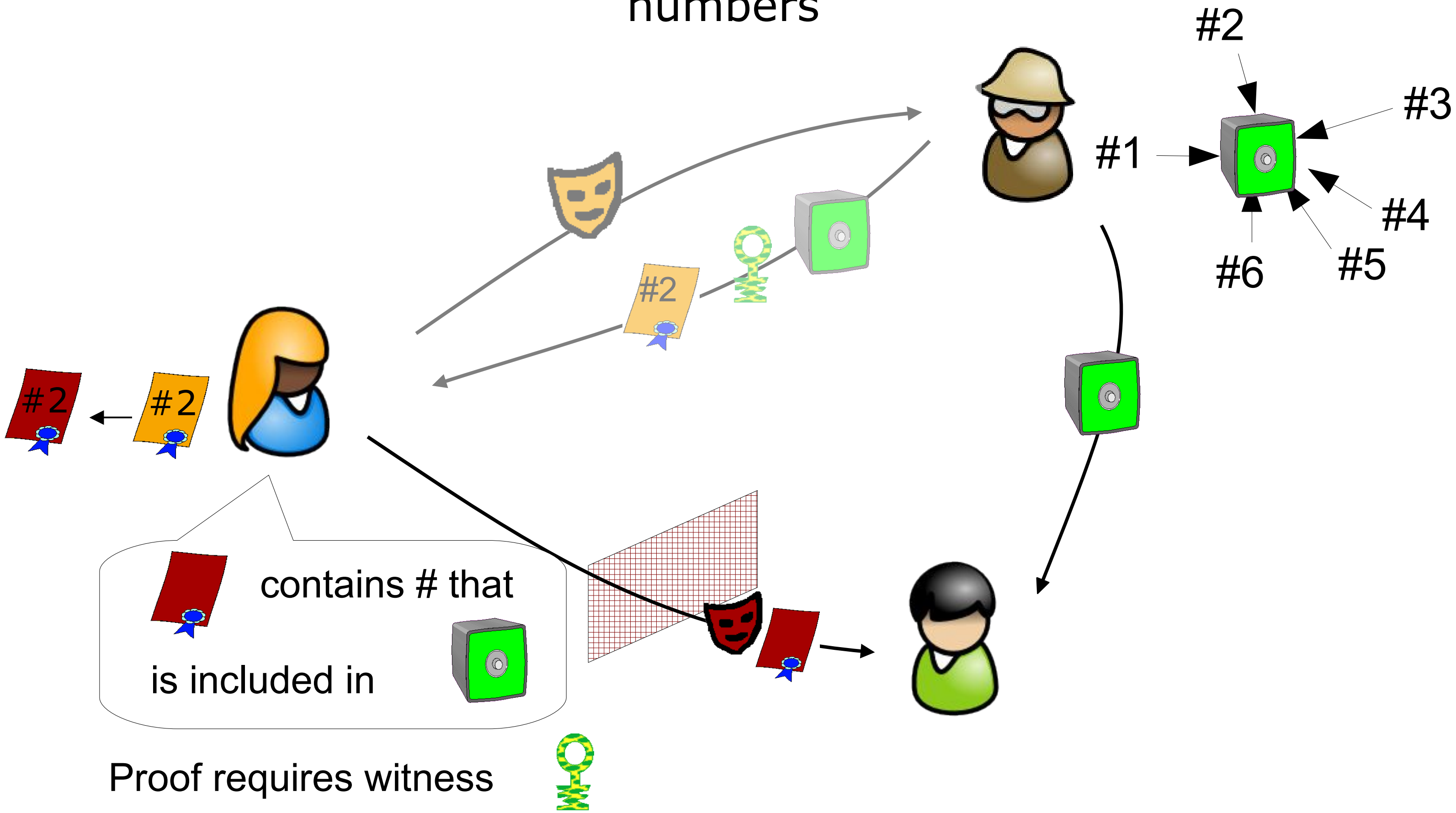
Sig(#i,#j) can be realised also with credential signature scheme, using different public key

Revocable Credentials: Third Solution

Using cryptographic accumulators:

credentials contain random serial number #

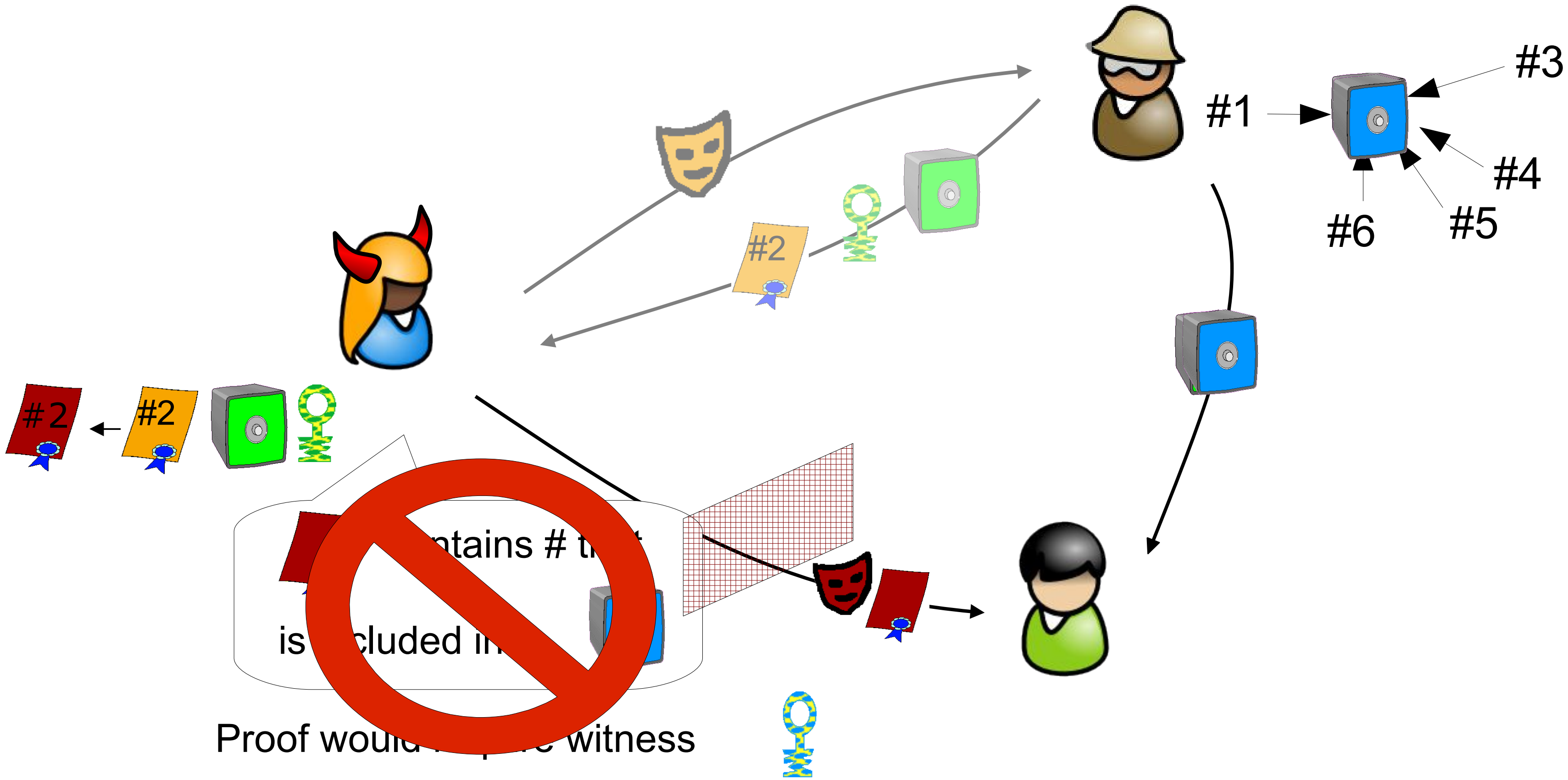
Issuer accumulates all "good" serial numbers



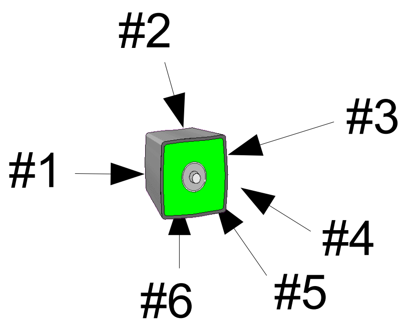
Revocable Credentials: Third Solution

Using cryptographic accumulators:

to revoke #2 issuer publishes new accumulator & new witnesses for unrevoked credentials



Revocable Credentials: Third Solution



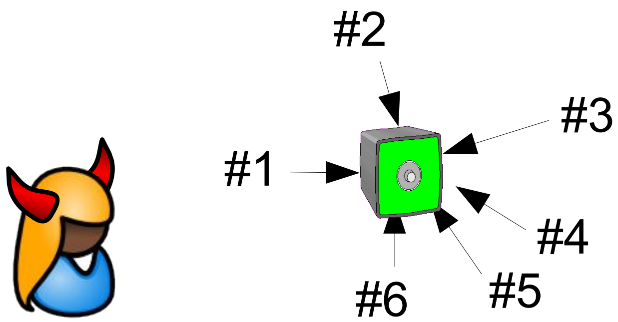
Using so-called cryptographic accumulators:

- Key setup: RSA modulus n, seed v
- Accumulate:
  - values are primes ei
  - accumulator value: $z = v^{\prod ei} \bmod n$
  - publish z and n
  - witness value x for ej : s.t. $z = x^{ej} \bmod n$
    can be computed as $x = v^{e1 \cdot \ldots \cdot ej-1 \cdot ej+1 \cdot \ldots \cdot ek} \bmod n$

- Show that your value e is contained in accumulator:
  - provide x for e
  - verifier checks $z = x^{e} \bmod n$

Security of accumulator: show that e s.t. $z = x^{e} \bmod n$ for e that is not contained in accumulator:

- For fixed e: Equivalent to RSA assumption
- Any e: Equivalent to Strong RSA assumption



Revocation: Each cert is associated with an e and each user gets witness x with certificate. But we still need:

- Efficient protocol to prove that committed value is contained in accumulator.
- Dynamic accumulator, i.e., ability to remove and add values to accumulator as certificates come and go.

- Prove that your key is in accumulator:
  - Commit to x:
    - choose random s and g and
    - compute $U1 = x\, h^{s}$, $U2 = g^{s}$ and reveal $U1$, $U2$, g
  - Run proof-protocol with verifier

$$PK\{(\varepsilon, \mu, \rho, \sigma, \xi, \delta) : d = c'^{\varepsilon} a_1^{\rho} a_2^{\mu} b^{\sigma} \pmod{n} \wedge z = U1^{\mu}(1/h)^{\xi} \pmod{n} \wedge 1 = U2^{\mu}(1/g)^{\xi} \pmod{n} \wedge U2 = g^{\delta} \pmod{n}\}$$

- Analysis
  - No information about x and e is revealed:
    - $(U1, U2)$ is a secure commitment to x
    - proof-protocol is zero-knowledge
  - Proof is indeed proving that e contained in the certificate is also contained in the accumulator:

a) $1 = U2^{\mu}(1/g)^{\xi} = (g^{\delta})^{\mu}\,(1/g)^{\xi} \pmod{n}$

$\Rightarrow \xi = \delta\,\mu$

b) $z = U1^{\mu}(1/h)^{\xi} = U1^{\mu}(1/h)^{\delta\,\mu} = (U1/h^{\delta})^{\mu} \pmod{n}$

c) $d = c'^{\varepsilon}\,a_1^{\rho}a_2^{\mu}\,b^{\sigma} \pmod{n}$

Dynamic Accumulator

- When a new user gets a certificate containing enew
  - Recall: $z = v^{\prod ei} \bmod n$
  - Thus: $z' = z^{enew} \bmod n$
  - But: then all witnesses are no longer valid, i.e., need to be updated $x' = x^{enew} \bmod n$

Dynamic Accumulator

- When a certificate containing erev revoked
  - Now $z' = v^{\prod ei} = z^{1/erev} \bmod n$
  - Witness:
    - Use Ext. Euclid to compute a and b
      s.t. $a\ eown + b\ erev = 1$
    - Now $x' = x^{b} z'^{a} \bmod n$
    - Why: $x'^{eown} = ((x^{b} z'^{a})^{eown})^{erev\ 1/erev} \bmod n$

$$= ((x^{b} z'^{a})^{eown\ erev})^{1/erev} \bmod n$$

$$= ((x^{eown})^{b\ erev} (z'^{erev})^{a\ eown})^{1/erev} \bmod n$$

$$= (z^{b\ erev} z^{a\ eown})^{1/erev} \bmod n$$

$$= z^{1/erev} \bmod n = z' \quad :-)$$

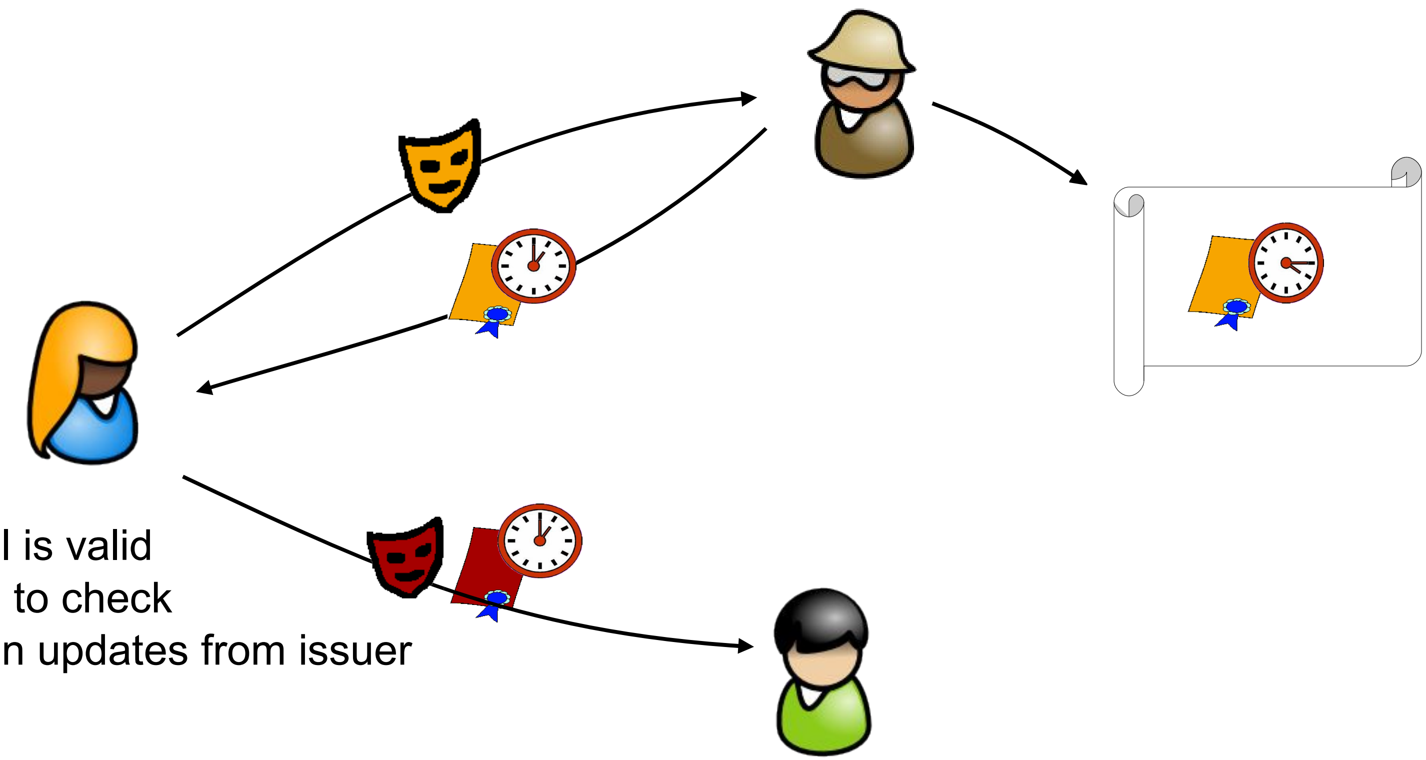
Dynamic Accumulator: in case the issuer knows the factorization of n

- When a new user gets a certificate containing $enew$
  - Recall: $z = v^{\Pi\, ei} \bmod n$
  - Actually v never occurs anywhere...
    so: $v' = v^{1/enew} \bmod n$ and $x = z^{1/enew} \bmod n$
  - Thus z needs not to be changed in case new member joins!

- Witnesses need to be recomputed upon revocation only!

Revocation: Zeroth Solution

Update of Credentials: encode validity time as attribute



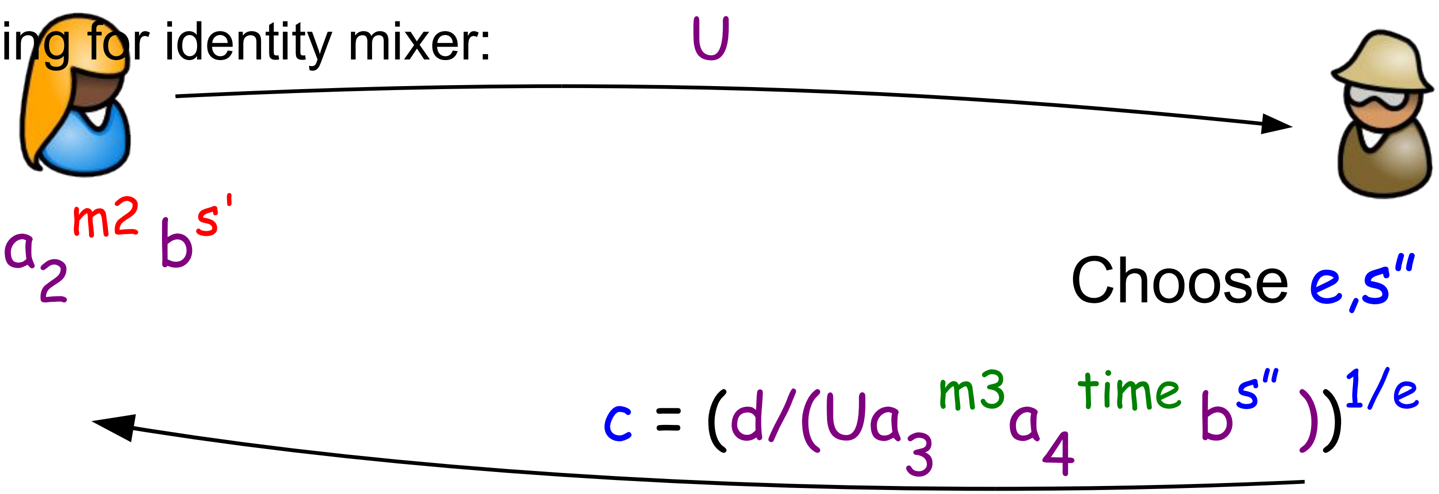
if credential is valid
→ no need to check
revocation updates from issuer

no additional effort for verifier

Re-issue certificates

(off-line – interaction might be too expensive)

Recall issuing for identity mixer:



$U := a_1^{m1} a_2^{m2} b^{s'}$

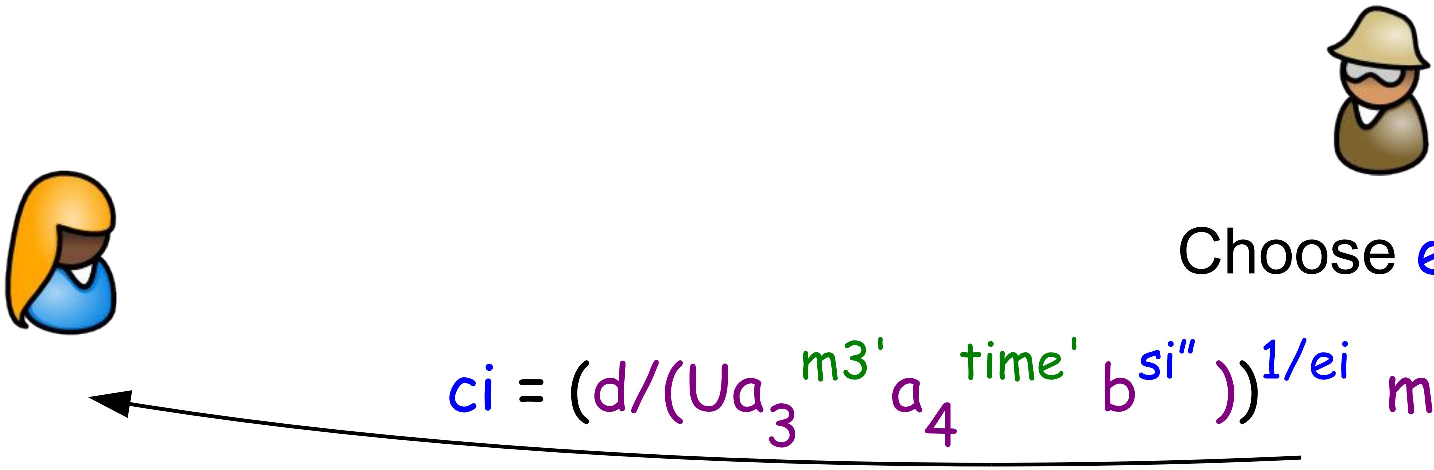
Choose e, s''

$c = (d/(U a_3^{m3} a_4^{time} b^{s''}))^{1/e} \bmod n$

(c, e, s'')

Re-issue certificates (off-line – interaction might be too expensive)

- Idea: just repeat last step for each new time time':



Choose ei,si”

$ci = (d/(Ua_3^{m3'} a_4^{time'} b^{si''}))^{1/ei} \bmod n$

(ci,ei,si”)

- Update information (ci,ei,si”) can be pushed to user by many different means

Conclusions

- Roadmap
  - Explain possibilities to engineers, policy makers etc
  - Usable prototypes
  - Provide transparency
  - Public infrastructure for privacy protection
  - Laws with teeth (encourage investment in privacy)

- Challenges
  - Internet services get paid with personal data (inverse incentive)
  - End users are not able to handle their data (user interfaces..)
  - Security technology typically invisible and hard to sell

- Towards a secure information society
  - Society changes quickly and gets shaped by technology
  - Consequences are hard to grasp (time will show...)
  - We must inform and engage in a dialog

Thank you!

- eMail: identity@zurich.ibm.com
- Links:
  - www.abc4trust.eu
  - www.futureID.eu
  - www.au2eu.eu
  - www.PrimeLife.eu
  - www.zurich.ibm.com/idemix
  - idemixdemo.zurich.ibm.com
- Code
  - github.com/p2abcengine & abc4trust.eu/idemix

References

- D. Chaum, J.-H. Evertse, and J. van de Graaf. *An improved protocol for demonstrating possession of discrete logarithms and some generalizations.* In EUROCRYPT ’87, vol. 304 of LNCS, pp. 127–141. Springer-Verlag, 1988.
- S. Brands. *Rapid demonstration of linear relations connected by boolean operators.* In EUROCRYPT ’97, vol. 1233 of LNCS, pp. 318–333. Springer Verlag, 1997.
- Mihir Bellare: Computational Number Theory http://www-cse.ucsd.edu/~mihir/cse207/w-cnt.pdf
- Camenisch, Lysanskaya: *Dynamic Accumulators and Applications to Efficient Revocation of Anonymous Credentials*. Crypto 2002, Lecture Notes in Computer Science, Springer Verlag.
- Ateniese, Song, Tsudik: Quasi-Efficient Revocation of Group Signatures. In Financial Cryptography 2002, Lecture Notes in Computer Science, Springer Verlag.
- Jan Camenisch, Natalie Casati, Thomas Gross, Victor Shoup: Credential Authenticated Identification and Key Exchange. CRYPTO 2010:255-276
- Jan Camenisch, Maria Dubovitskaya, Gregory Neven: Oblivious transfer with access control. ACM Conference on Computer and Communications Security 2009: 131-140

- Ateniese, Song, Tsudik: Quasi-Efficient Revocation of Group Signatures. In Financial Cryptography 2002, Lecture Notes in Computer Science, Springer Verlag.

- M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko: *The One-More-RSA-Inversion Problems and the Security of Chaum's Blind Signature Scheme.* Journal of Cryptology, Volume 16, Number 3. Pages 185 -215, Springer-Verlag, 2003.
- E. Bangerter, J. Camenisch and A. Lyskanskaya: A Cryptographic Framework for the Controlled Release Of Certified Data. In Twelfth International Workshop on Security Protocols 2004. www.zurich.ibm.com/~jca/publications
- Stefan Brands: Untraceable Off-line Cash in Wallets With Observers: In Advances in Cryptology – CRYPTO '93. Springer Verlag, 1993.

References

- J. Camenisch and A. Lyskanskaya: Efficient Non-transferable Anonymous Multi-show Credential System with Optional Anonymity Revocation. www.zurich.ibm.com/~jca/publications
- David Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. In Communications of the ACM, Vol. 24 No. 2, pp. 84—88, 1981.
- David Chaum: Blind Signatures for Untraceable Payments. In Advances in Cryptology – Proceedings of CRYPTO '82, 1983.
- David Chaum: Security Without Identification: Transaction Systems to Make Big Brother obsolete: in Communications of the ACM, Vol. 28 No. 10, 1985.
- Camenisch, Shoup: Practical Verifiable Encryption and Decryption of Discrete Logarithms. CRYPTO 2003: 126-144
- Victor Shoup: A computational introduction to Number Theory and Algebra. Available from: http://www.shoup.net/ntb/
- D. Chaum: *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms.* In Communications of the ACM.
- D. Chaum: *The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceabilit*y. Journal of Cryptology, 1988.
- J. Camenisch and V. Shoup: *Practical Verifiable Encryption and Decryption of Discrete Logarithms.* In Advances in Cryptology - CRYPTO 2003.
- T. ElGamal: *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms.* In Advances in Cryptology - CRYPTO '84.