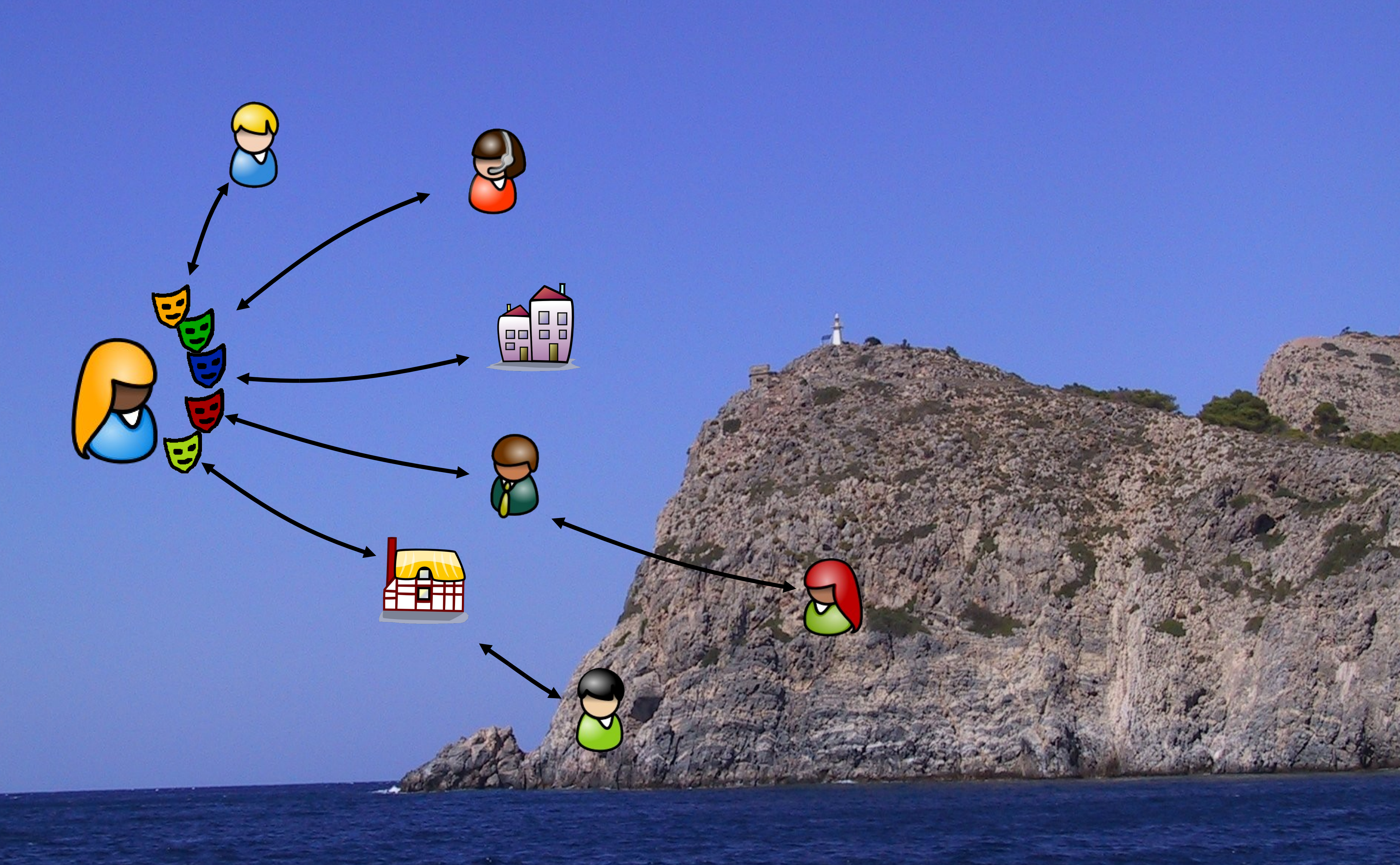
Tutorial Identity Mixer & Overview ABC4Trust

Jan Camenisch
IBM Research – Zurich

vision:
a secure and privacy-protecting e-world



Solutions today

Online security & trust today:

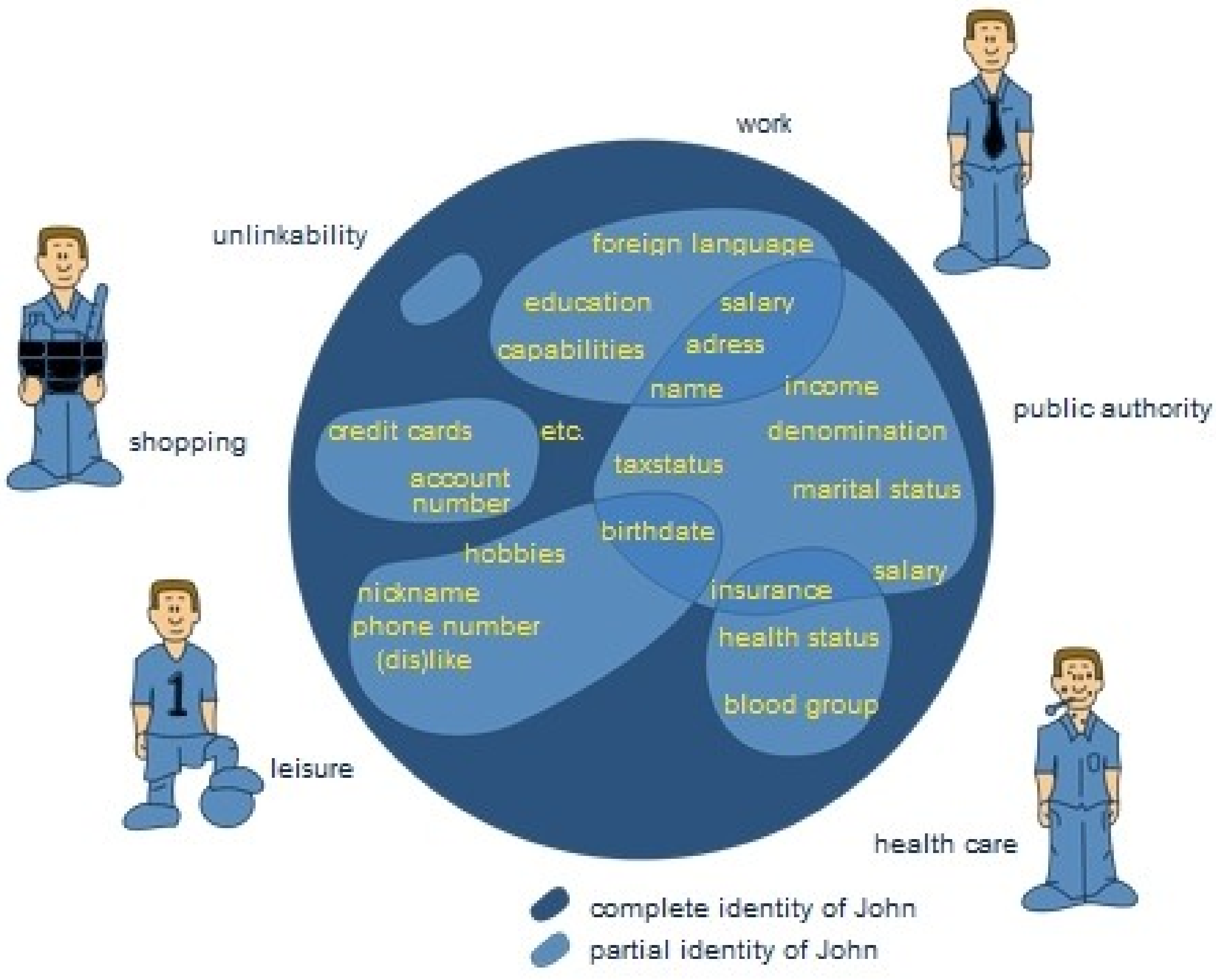
- SSL/TLS does encryption and server-side authentication
- Client-side authentication by username-password
- Mostly self-claimed attributes

Alternative approaches exist

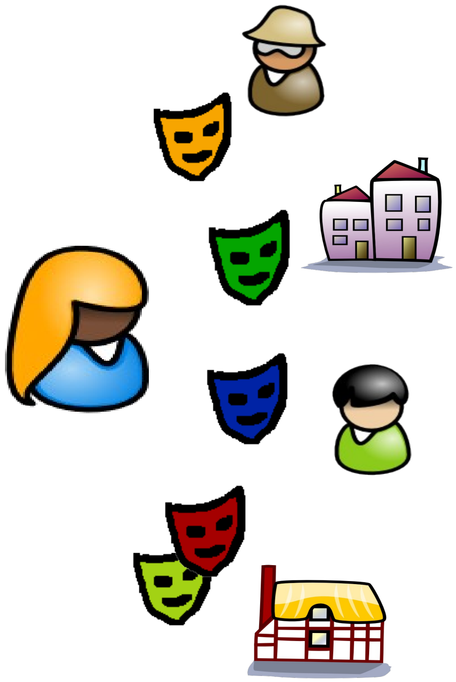
- e.g., SAML, OpenID, Facebook Connect, X.509…

… but have privacy and security issues

what is an identity?



Identities – need to be managed



- ID: set of attributes shared w/ someone
  - attributes are not static: user & party can add
- ID Management: make ID useful
  - ID comes w/ authentication means
  - transport attributes between parties
- Privacy & Security:
  - user in control of transport
    - Policies

      request definition

      allowed usage (audience)
    - polices authored by user or party

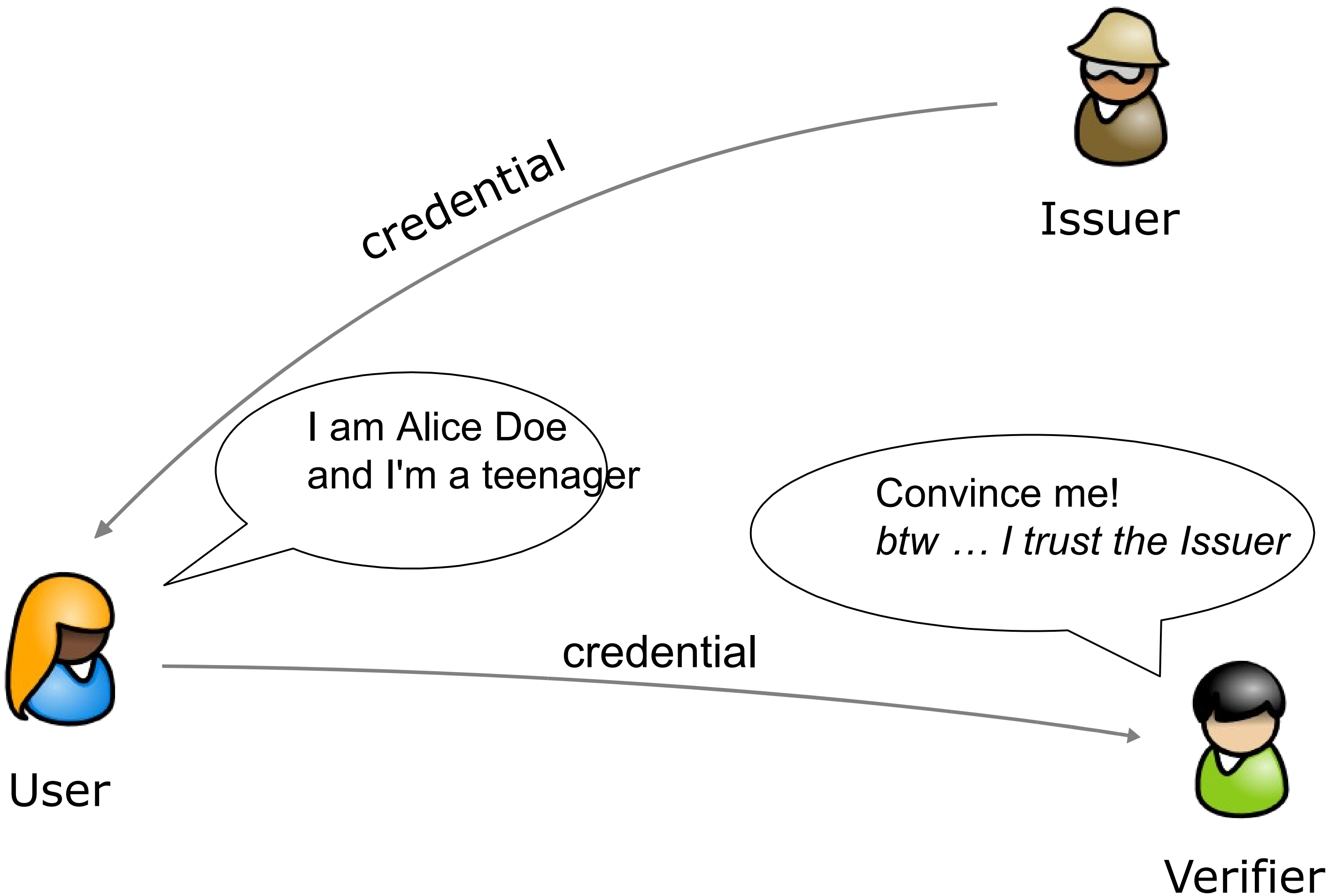
      E-commerce

      social networks, delegation
  - policies are enforced technically (Security....)
    - No side information is revealed
    - anonymous credentials, encryption, etc

Example Scenario: Access to a Teenage Chat Room

- Goal: Only teenagers in the chat room
- Solution 1:
  Use electronic identity cards
- eID
  - Use digital signatures to issue certificate
  - Show certificate to chatroom

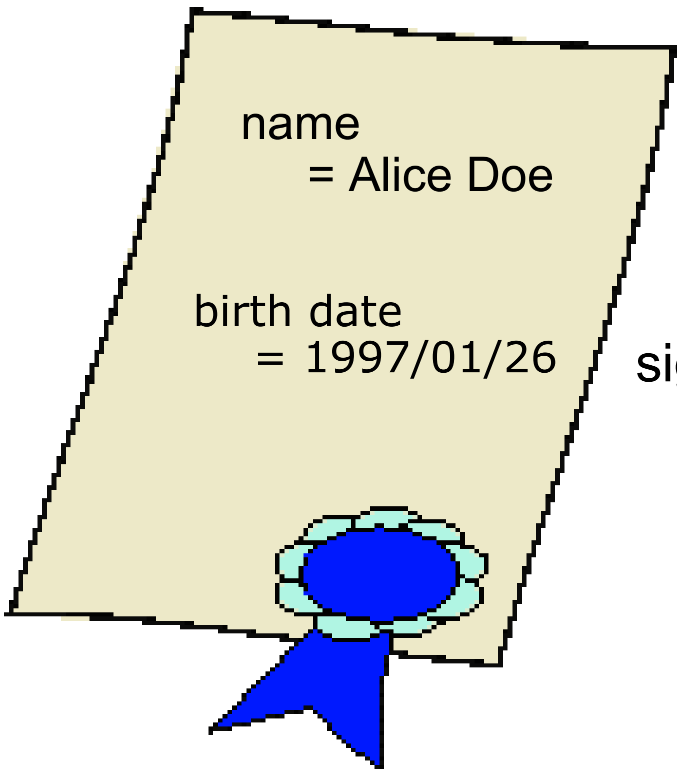
Solution1: Traditional PKI



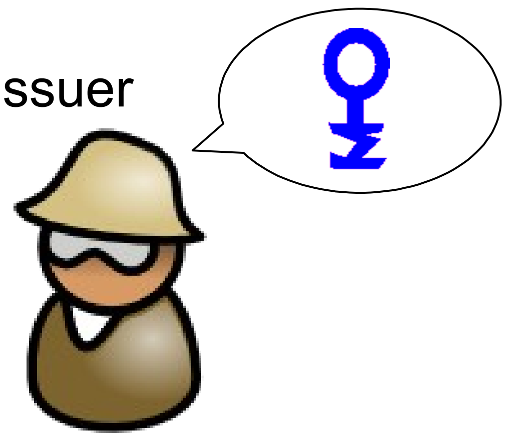
Solution1: Traditional PKI

credential / certificate

- signed list of attribute-value pairs



signed by the issuer



Solution1: Traditional PKI

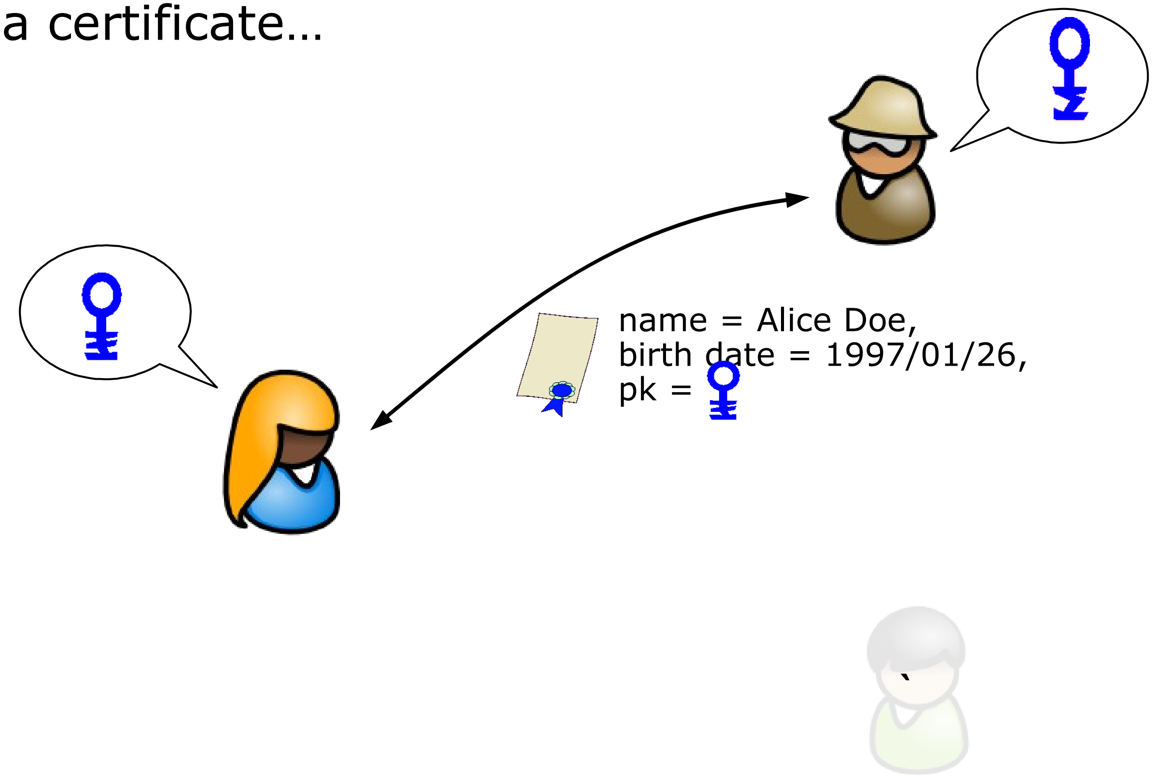
e.g., X.509 certificates

In the beginning…

Solution1: Traditional PKI

e.g., X.509 certificates

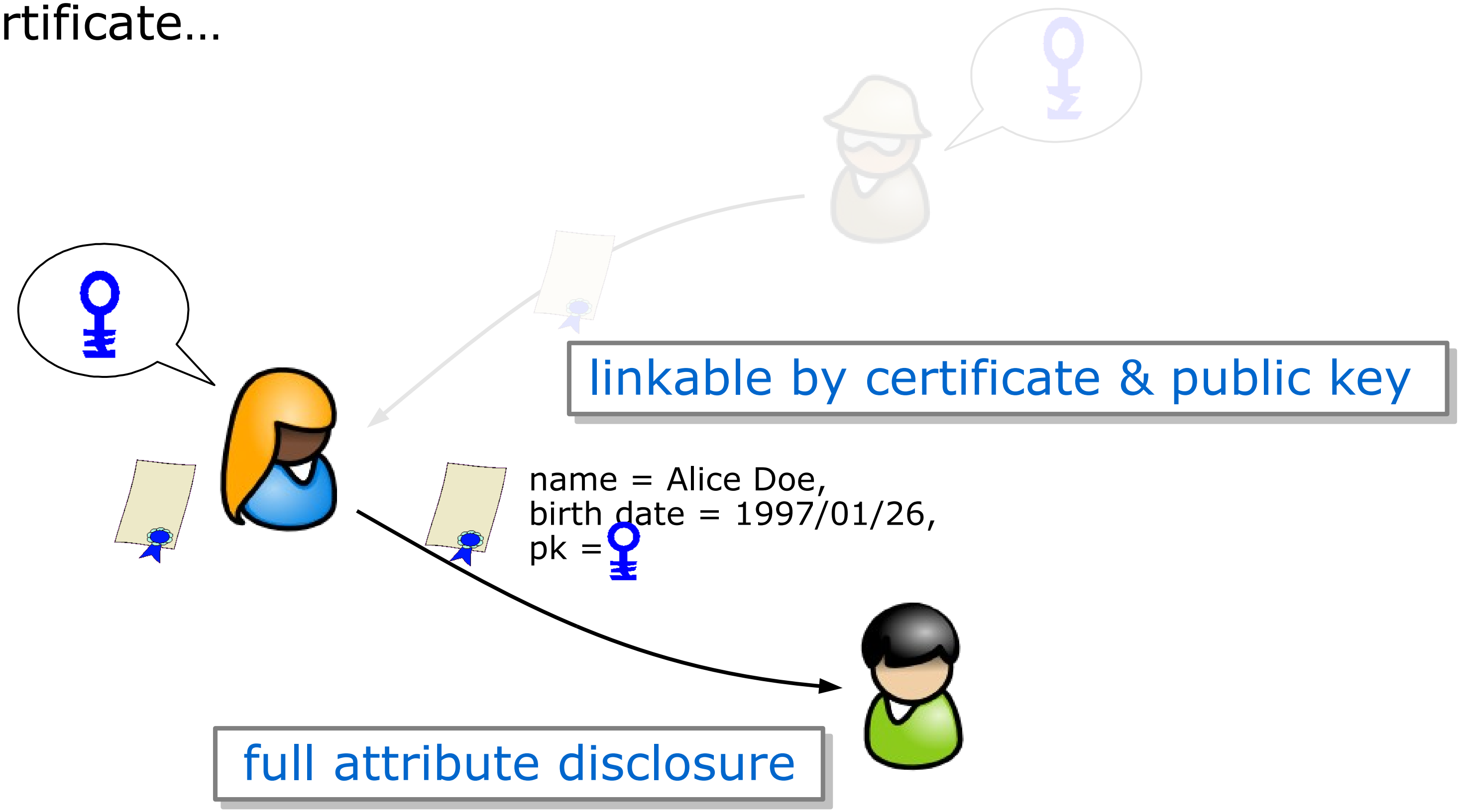
Obtaining a certificate…



Solution1: Traditional PKI

e.g., X.509 certificates

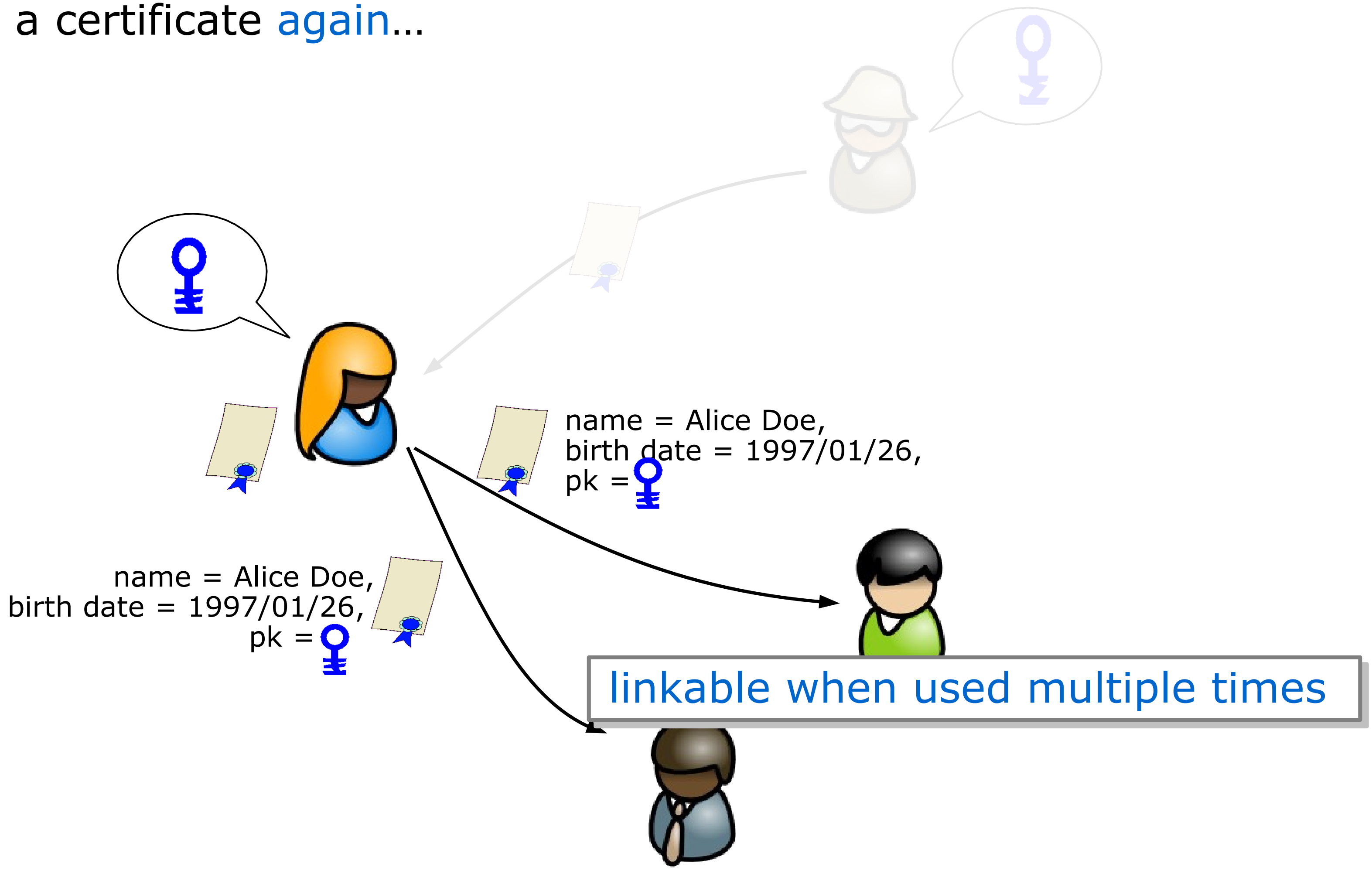
Using a certificate…



Solution1: Traditional PKI

e.g., X.509 certificates

Using a certificate again…



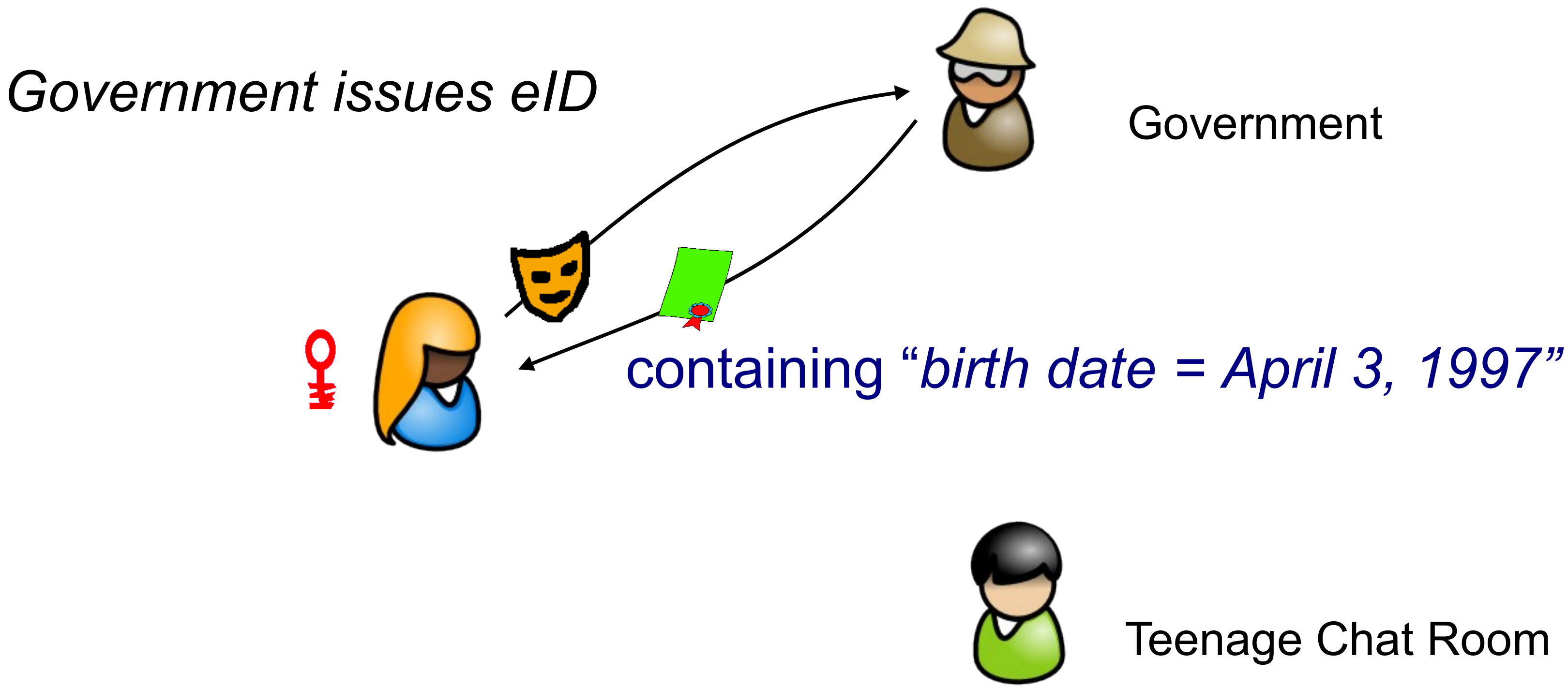
Access to a teenage chatroom

- Goal: Only teenagers in the chat room

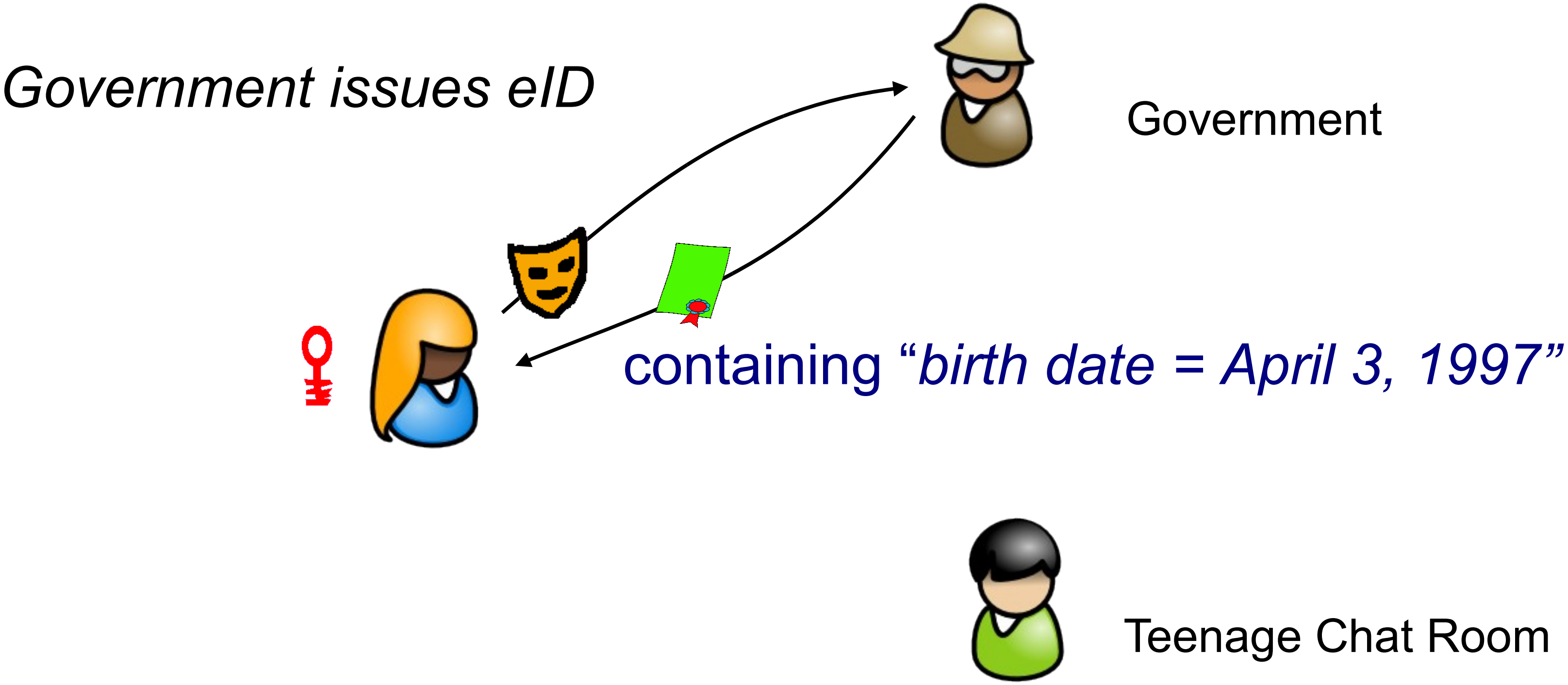
- Solution 1:
  Use electronic identity cards
- Problem:
  Chat room gets much more information

- Solution 2:
  Use anonymous credential to *prove just* age

Solution 2: private credentials

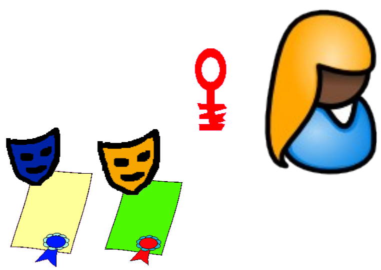


Solution 2: private credentials

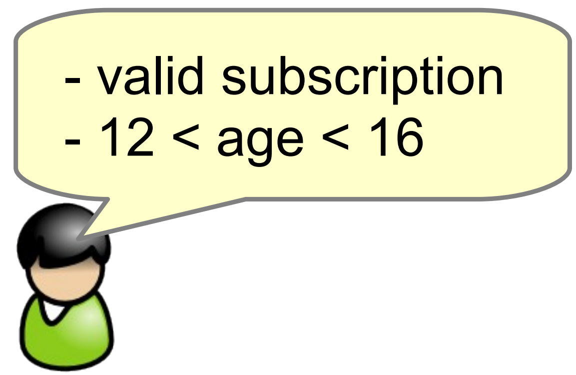


Solution 2: private credentials

Teenager wants to chat

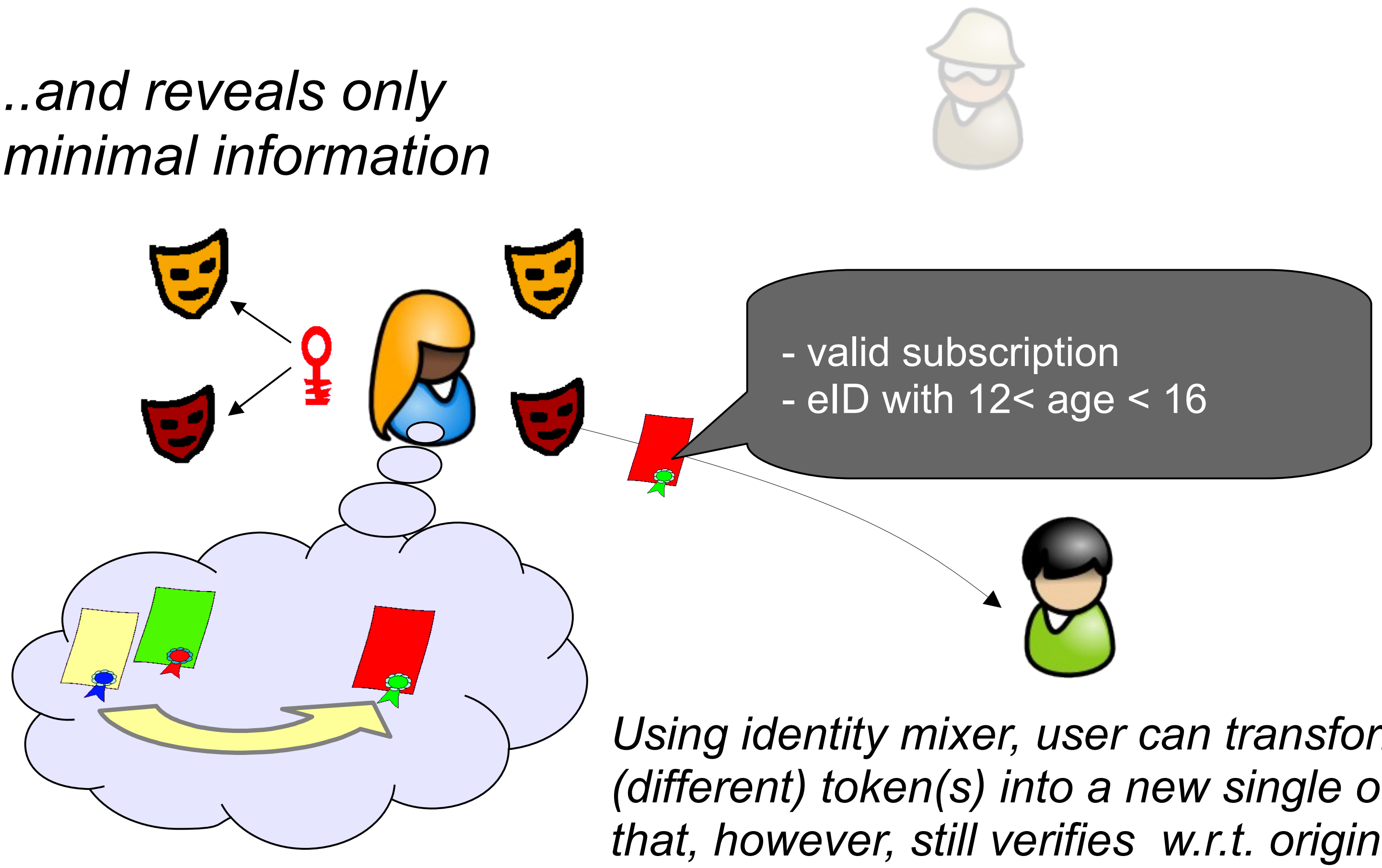


goes off-line



Solution 2: private credentials

..and reveals only minimal information



Using identity mixer, user can transform (different) token(s) into a new single one that, however, still verifies w.r.t. original signers' public keys.

Access to a Teenage Chat Room

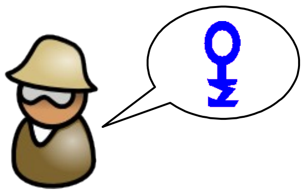
- Goal: Only teenagers in the chat room

- Solution 1:
  Use electronic identity cards

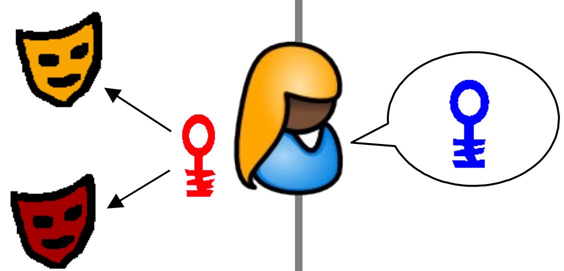
- Solution 2:
  Use anonymous credential to prove just age
- Problem: Abuse?
  - Parent could use teenager's card...
  - Cannot investigate...

- Can handle these; but compare other solutions

Anonymous credentials vs. classical certificates

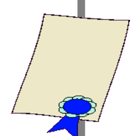


each issuer has public key of signature scheme



each user has a single secret key but many public keys

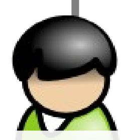
each user has a single public key



certificate = signature on user's secret key + attributes

attributes might be hidden from issuer

certificate = signature on user's public key + attributes



using certificate

prove knowledge of certificate

- prove that different certificates contain same secret key
- selectively reveal attributes

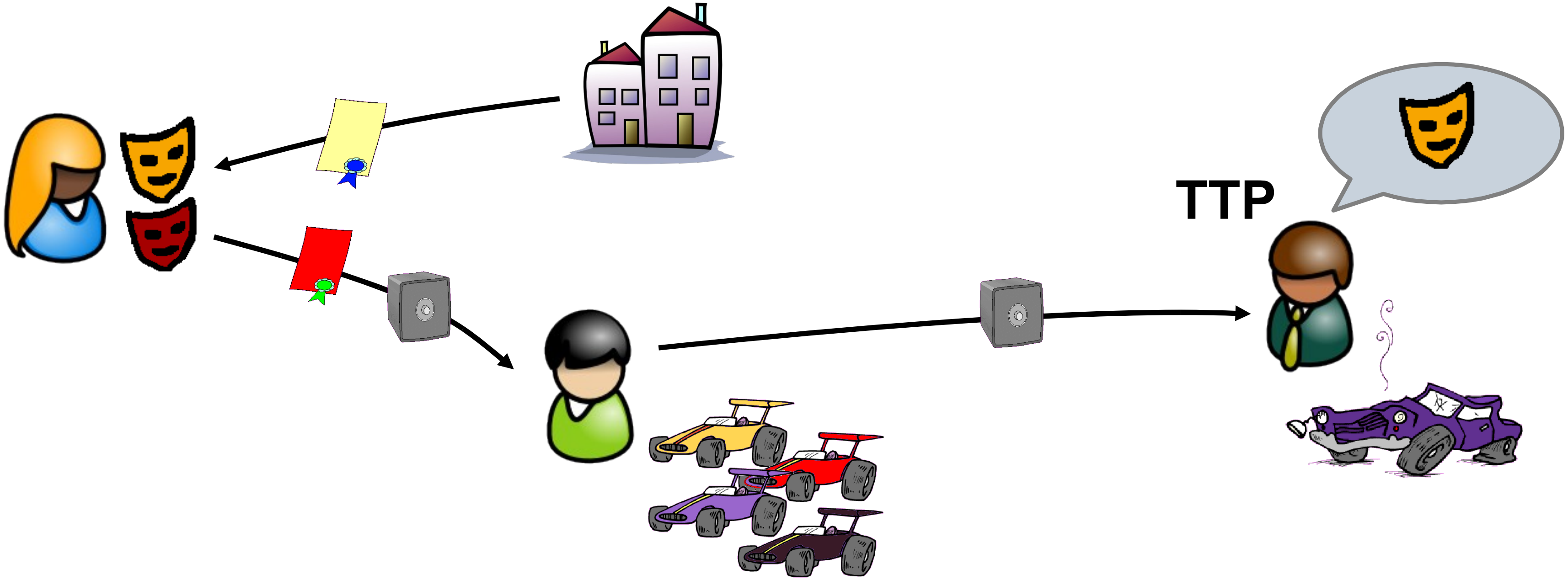
reveal certificate

Access to a Teenage Chat Room

Demo Time

Extended Features

Other Properties: Attribute Escrow (Opt-In)

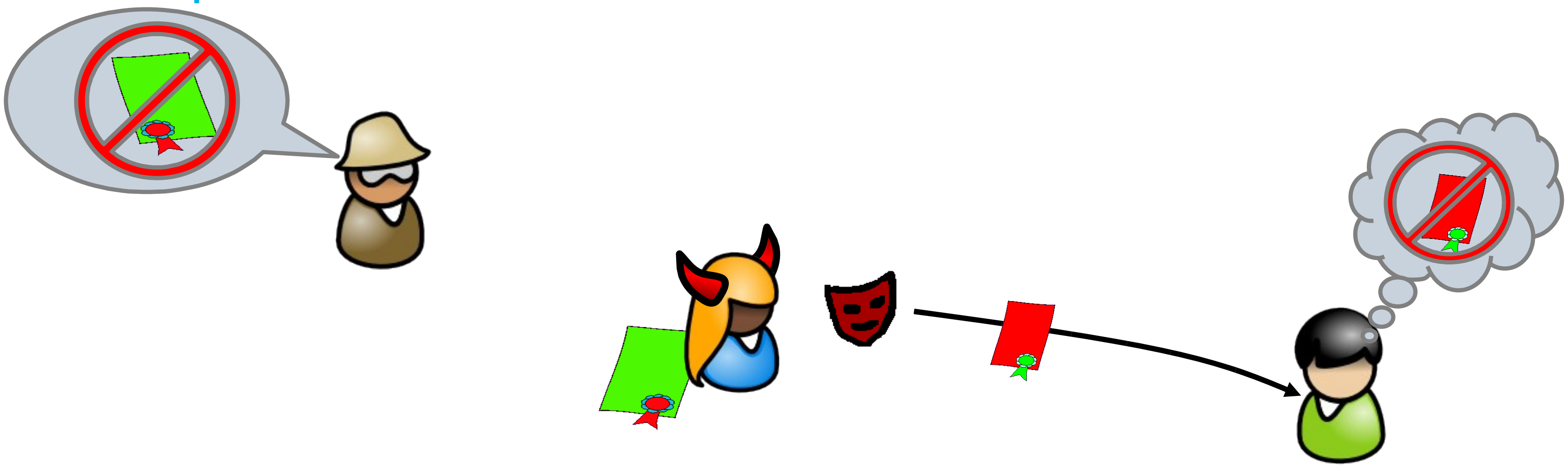


If car is broken: ID with insurance needs be retrieved

Can verifiably encrypt any certified attribute *(optional)*

TTP is off-line & can be distributed to lessen trust

Other Properties: Revocation

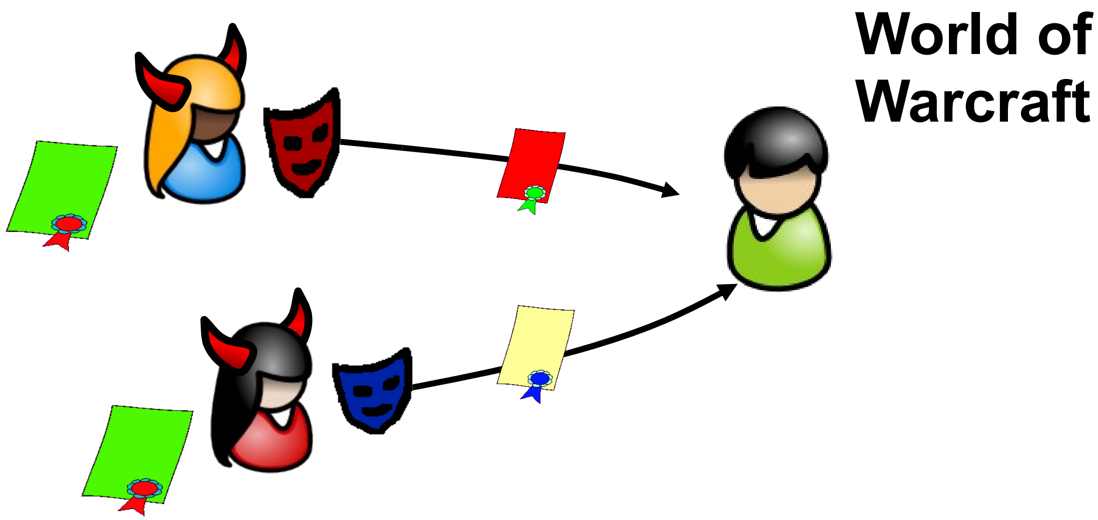


If Alice was speeding, license needs to be revoked!

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

Other Properties: Cheating Prevention



Limits of anonymity possible *(optional)*:

If Alice and Eve are on-line together 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)

Privacy Preserving Access Control [CDN09]

DNA Database

Simple case: DB learns not who accesses DB

Better: Oblivious Access to Database (OT with AC)

- Server must not learn *who* accesses
- *which* record
- Still, Alice can access only records she is *authorized* for

Suitable signatures scheme

Digital Signature Schemes 4 Privacy

- Sign blocks of messages m1, … , mk
- Compatible with proof protocols
- Some known schemes:
  - Brands/U-Prove (Discrete Log/Blind Signature)
  - Camenisch-Lysyanskaya (Strong RSA)
  - Camenisch-Lysyanskaya (Bilinear Maps; LRSW, q-SDH)
  - ....a number of others, but not really practical yet
    - P-Signatures – Belinkiy et al. (q-SDH)
    - Lattice-based ones (Gordon et al.)

CL Signature Scheme

SRSA Variant

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

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

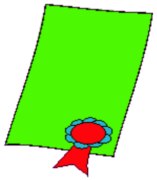
Signature Scheme based on SRSA [CL01]

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 $e > 2^\ell$ and integer $s \approx n$
- compute c such that



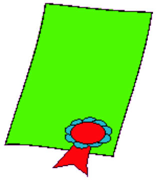
$$d = a_1^{m1} \cdot \ldots \cdot a_k^{mk} \, b^s \, c^e \mod n$$

- signature is (c,e,s)

Signature Scheme based on SRSA [CL01]

A signature (c,e,s) on messages $m1, ..., mk$ is valid iff:

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



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

Schnorr Protocol

(also called signature of knowledge)

From Protocol To Signature

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

Prover:

random $r \in Z_q$

$t := g^r$ → t

← c — random $c \in \{0,1\}^l$

$s := r - cx \quad (q)$ → s

Verifier:

$t = g^s y^c$

From Protocol To Signature

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

Signing a message m:

- chose random $r \in Z_q$ and
- compute $(c,s) := (H(g^r||m), r - cx \quad (q))$

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

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

Security:

- Discrete Logarithm Assumption holds
- Hash function $H(.)$ behaves as a random oracle.

Zero Knowledge Proofs

Non-interactive (Fiat-Shamir heuristic):

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

Logical combinations:

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

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

Intervals and groups of different order (under SRSA):

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

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

U-Prove Signature Scheme

U-Prove seen as a Signature Scheme [Brands93,..]

Public key of signer: Group $G = \langle g \rangle$ and $g_0, g_1, \ldots, g_k, g_{k+1}, y = g^x$

Secret key: x

To sign k messages $m1, \ldots, mk \in Zq$:

I. Choose random $a \in Zq$

II. Compute $h = (g_0\, g_1^{m1} \cdot \ldots \cdot g_k^{mk}\, g_{k+1}^{ctype})^a$ and $z = h^x$

III. Compute $(c, s) = SPK\{(x): y = g^x \wedge z = h^x\}(z,h)$

IV. Signature is $(a, h, z, (c,s))$

where ctype is a fixed non-zero value derived from public key and anapplication identifier.

Note: a could be fixed value if we are only interested in sign. scheme.

U-Prove seen as a Signature Scheme [Brands93,..]

Public key of signer: Group $G = \langle g \rangle$ and $g_0, g_1, \ldots, g_k, g_{k+1}, y$

To verify signature $(a, h, z, (c, s))$ on k messages $m1, \ldots, mk \in Zq$:

I. check $h = (g_0\, g_1^{m1} \cdot \ldots \cdot g_k^{mk}\, g_{k+1}^{ctype})^a$

II. verify $(c, s) = SPK\{(x): y = g^x \wedge z = h^x\}(z,h)$

Security:

- requires DL to be hard ...

- … but assumption is that scheme is secure (no reduction known)

Alternative Signature Schemes

Signature schemes that follow the same paradigm

- SRSA-based (just presented)
- Bi-linear maps based: LRSW-Assumption
- Bi-linear maps based: q-SDH-Assumption

.... Brands DL-based scheme (uProve)

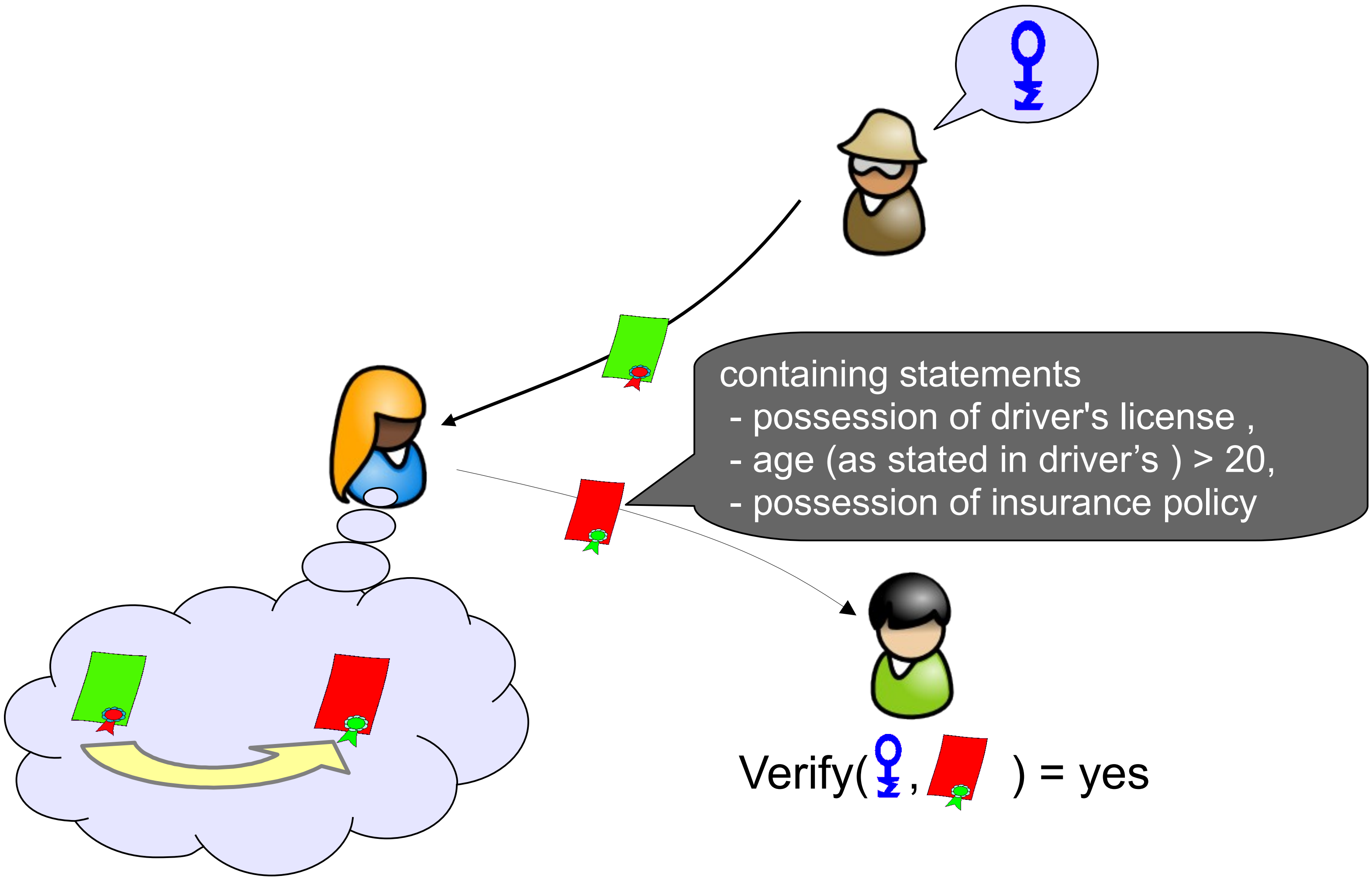
- Blind signatures (use once)
- No security proofs

… a number of schemes geared towards GS-proofs

- not efficient enough yet

Proving possession of a signature

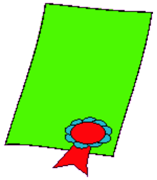
Recall Goal...


containing statements
- possession of driver's license ,
- age (as stated in driver's) > 20,
- possession of insurance policy
Verify(,) = yes

Recall Verification of Signature

A signature (c,e,s) on messages $m1, \ldots, mk$ is valid iff:

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



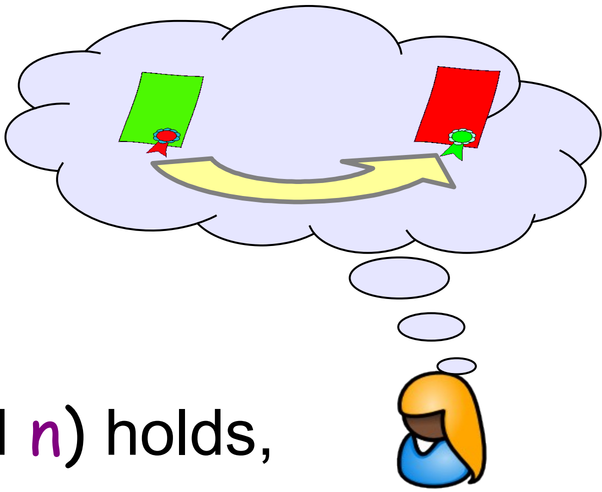
Thus to prove knowledge of values

$m1, \ldots, mk, e, s, c$

such that the above equations hold.

Problem: c is not an exponent...

Proof of Knowledge of a CL Signature



Solution randomize c :

- Let $c' = c\, b^{s'} \bmod n$ with random s'
- then $d = c'^{e}\, a_1^{m1} \cdot \ldots \cdot a_k^{mk}\, b^{s^*} \pmod n$ holds,

  i.e., (c', e, s^*) is a also a valid signature!

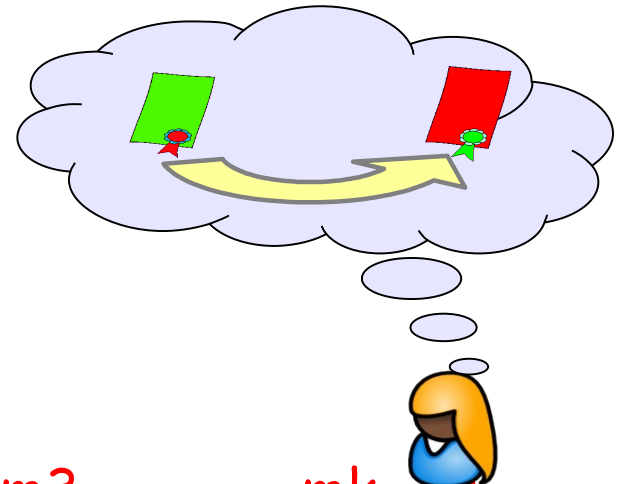
Therefore, to prove knowledge of signature on hidden msgs:

- provide c'
- $PK\{(e, m1, \ldots, mk, s) : \quad d = c'^{e} a_1^{m1} \cdot \ldots \cdot a_k^{mk} b^{s}$

  $\wedge \quad mi \in \{0,1\}^{\ell} \quad \wedge \quad e \in 2^{\ell+1} \pm \{0,1\}^{\ell} \quad \}$

Proof of Knowledge of a CL Signature

.....revealing some of the messages



- Randomise, $c' = c\, b^{s'} \bmod n$
- provide c'
- PK{(e, m3, ..., mk, s) :

$$d/(a_1^{m1}\, a_2^{m2}) = c'^{e}\, a_3^{m3} \cdot \ldots \cdot a_k^{mk}\, b^{s}$$

$$\wedge \;\; mi \in \{0,1\}^{\ell} \;\; \wedge \;\; e \in 2^{\ell+1} \pm \{0,1\}^{\ell} \;\; \}$$

Proving Possession of a U-Prove Signature

Presentation of U-Prove Tokens

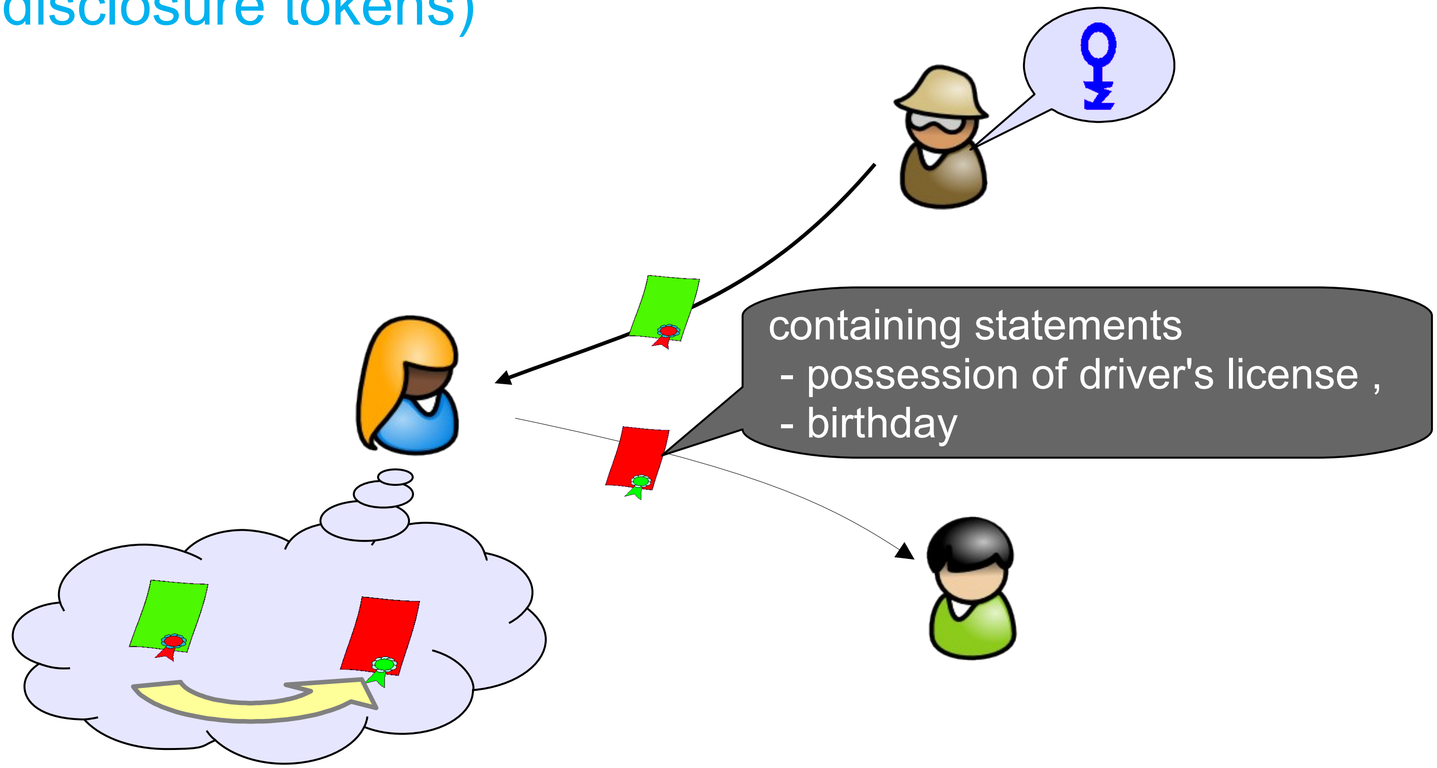
Recall: To verify signature (a, h,z,(c,s)) on k messages m1, ..., mk ∈ Zq:

- Check $h = (g_0\, g_1^{m1} \cdot \ldots \cdot g_k^{mk}\, g_{k+1}^{ctype})^a$
- verify $(c, s) = SPK\{(x): y = g^x \wedge z = h^x\}(z,h)$

To prove possession of signature (a, h,z,(c,s)) on messages mi :

- $(c', s\_i) = SPK\{(mi, a): 1/(g_0\, g_{k+1}^{ctype}) = h^{-1/a}\, g_1^{m1} \cdot \ldots \cdot g_k^{mk}\}$
- $(c, s) = SPK\{(x): y = g^x \wedge z = h^x\}(z,h)$

So we can already do anonymous tokens (or minimal disclosure tokens)



Includes:

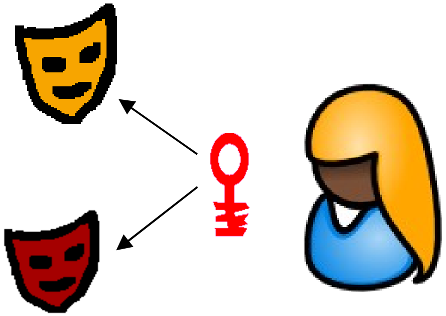
– On-line e-cash (merchant checks validity of cash w/bank)

But not

– predicates over attribute, revocation, tracing, etc,....

Pseudonyms

(Cryptographic) Pseudonyms



Algebraic Setting: Group $G = \langle g \rangle$ of order q.

Pseudonyms:

- Secret identity: $sk \in Zq$.
- Pseudonym: pick random $r \in Zq$ and compute $P = g^{sk}h^{r}$.
- Scope exclusive pseudonym:
  - let $g_d = H(scope)$. Then compute $P = g_d^{sk}$.
    Thus domain pseudonym as unique (per secret identity)

Security:

- Pseudonyms are perfectly unlinkeable.
- Domain pseudonyms are unlinkeable provided
  - Discrete logarithm assumption holds and
  - $H(scope)$ is a random function.

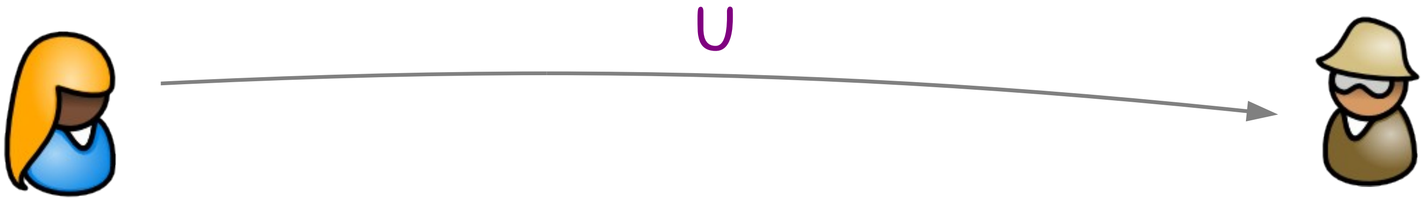
Issuing Credentials Extended

- To Pseudonyms
- Hidden Messages

CL Signature Scheme

Issuing a credential to hidden messages (idemix)

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



$PK\{(m1, m2, s') : U = a_1^{m1} a_2^{m2} b^{s'} \wedge mi \in \{0,1\}^{\ell} \}$

Issuing a credential to hidden messages (idemix)

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

U

Choose e, s''

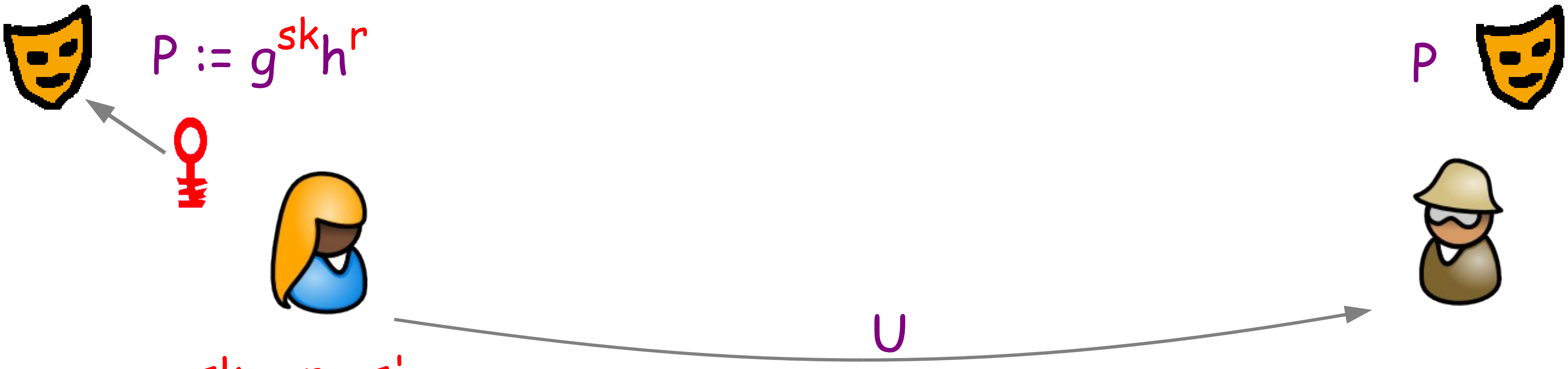
$c = (d/(U a_3^{m3} b^{s''}))^{1/e} \bmod n$

(c, e, s'')

$$d = a_1^{m1} a_2^{m2} a_3^{m3} b^{s'' + s'} c^e \pmod{n}$$

Issuing a Credential to a Pseudonym (idemix)

Issuing a Credential to a Pseudonym (idemix)



$P := g^{sk}h^{r}$

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

$PK\{(sk, r, s') : P = g^{sk}h^{r} \wedge U = a_1^{sk} a_2^{r} b^{s'} \wedge sk, r \in \{0,1\}^{\ell}\}$

.... and then issue credential just as before

Issuing on Hidden & Committed Attributes

$$Com := g_1^{m1} g_2^{m2} h^r$$

$$U := a_1^{sk} a_2^{m1} a_3^{m2} b^{s'}$$

Com

U

$$PK\{(sk,m1,m2,r,s') : Com = g_1^{m1} g_2^{m2} h^r \wedge U = a_1^{sk} a_2^{m1} a_3^{m2} b^{s'} \wedge sk,m1,m2 \in \{0,1\}^{\ell}\}$$

Example use case: issue credential on last name

- Commit to last name
- Prove correctness using Government credential
- Get new credential issued

U-Prove Signature Scheme

… Issuing U-Prove Signatures/Credentials

random $a \in Z_q$

$h = h'^{a}$

$$h' = g_0\, g_1^{m1} \cdot \ldots \cdot g_k^{mk}\, g_{k+1}^{ctype}$$

a →

$$PK\{(x): y = g^{x} \wedge z' = h'^{x}\}$$

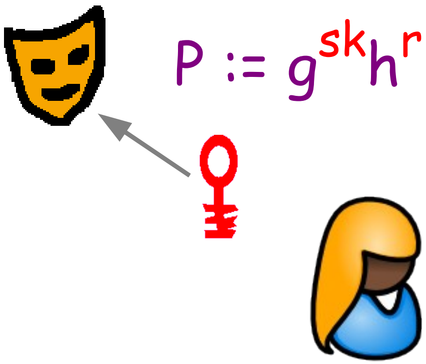
← x

$$SPK\{(x): y = g^{x} \wedge z = h^{x}\}$$

(z,h)

- A kind of blind signature scheme to ensure privacy
- Security
  - Based on blind signatures, but not quite
  - Does *not* reduce to assuming U-Prove Signatures are secure

Issuing on Pseudonyms: essentially the same....



$P := g^{sk} h^{r}$

P

U

$PK\{(sk, r, s') : P = g^{sk} h^{r} \wedge U = g_1^{sk} g_2^{r} g_k^{s'}\}$

$U := g_1^{sk} g_2^{r} g_k^{s'}$

$h' = U \cdot g_0 \cdot g_3 \cdot \ldots \cdot g_{k-1}^{mk-1} g_{k+1}^{ctype}$

random $a \in Zq$

$h = h'^{a}$

a

x

$PK\{(x): y = g^{x} \wedge z' = h'^{x}\}$

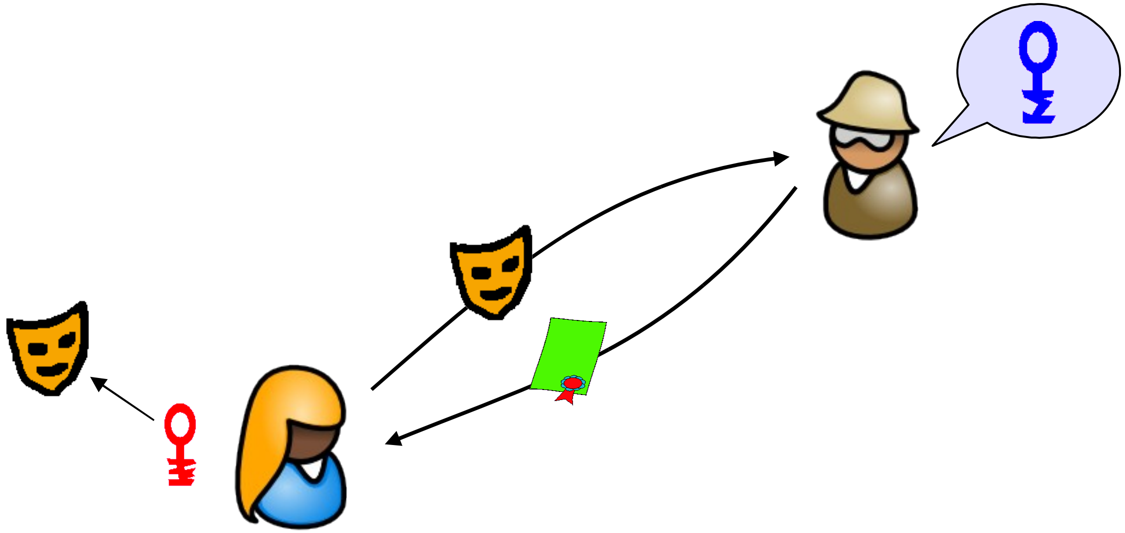
$SPK\{(x): y = g^{x} \wedge z = h^{x}\}(z,h)$

Example: Polling

Polling: Scenario and Requirements

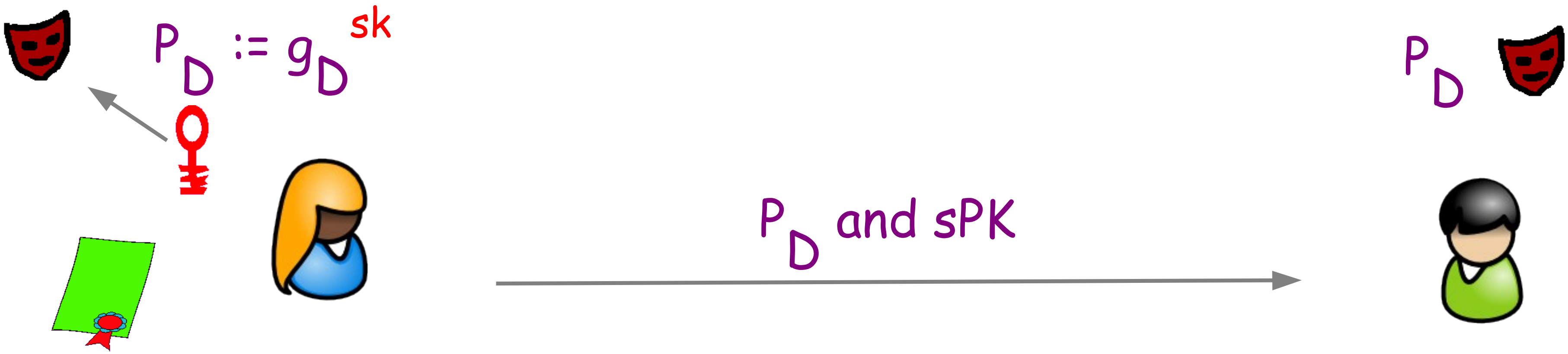
- User obtain credential that they are eligible under ID
- User can voice opinion anonymously
  - Different polls must not be linkable
- User can do so only once per poll
  - or if they do multiple times, only one voice counts – first, last?

Polling: Solution



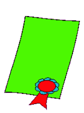
- User generates pseudonym (Id for registration)
- User obtains credential on pseudonym stating that she is eligible for polls
- Credential can contain attributes about her

Proving a credential w.r.t. a Pseudonym



$SPK\{(sk, r, m2, .., mk):$

$P_D = g_D^{sk} \wedge$



$d = c'^{e} a_1^{sk} \cdot a_2^{r} \cdot a_3^{m3} \cdot \ldots \cdot a_k^{mk} b^{s} \wedge$

$sk, r, mi \in \{0,1\}^{\ell} \wedge e \in 2^{\ell+1} \pm \{0,1\}^{\ell}\}$

(opinion)

Scope exclusive pseudonym different for each poll, e.g., $g_D = f(poll)$

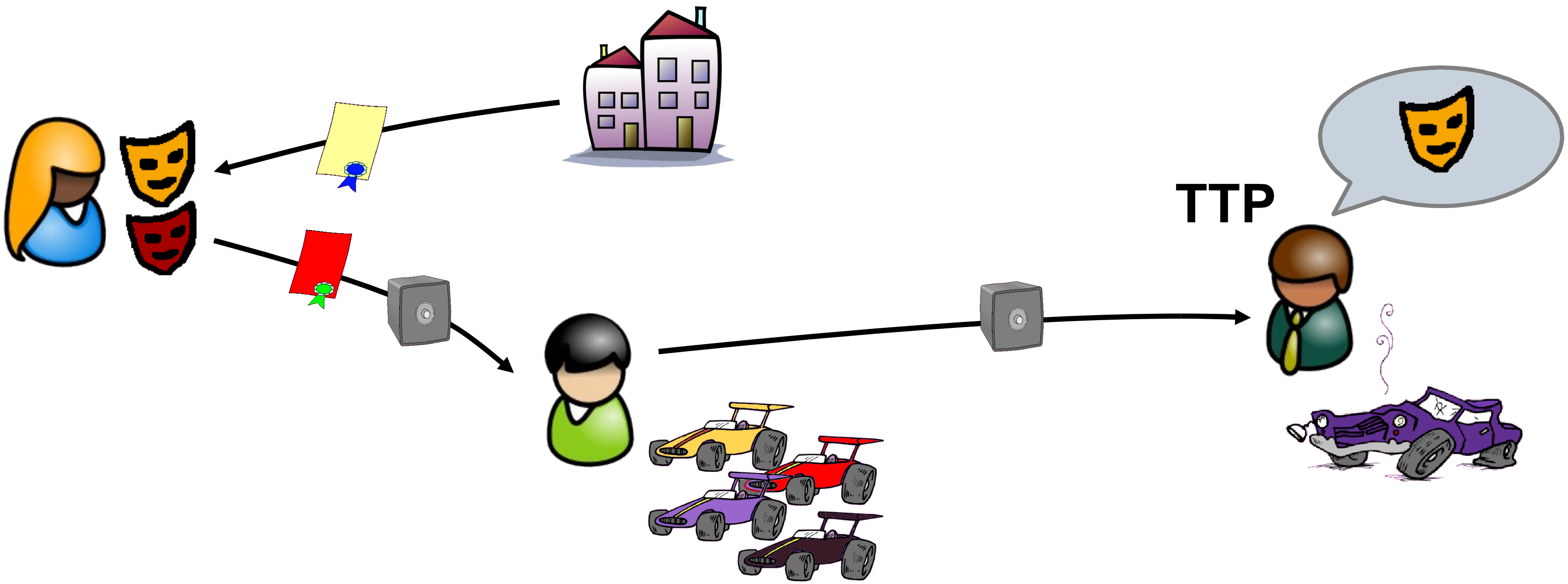
Polling: Extension

Can require several credentials, e.g.,

- User registers with university and obtains student credential
- User takes course and exam and gets a second credential on different pseudonym
- When polling, user proves w.r.t. domain pseudonym possession of
  - Student credential
  - Course credential

Verifiable Encryption

Motivation: Tracing & Attribute Escrow (Opt-In)

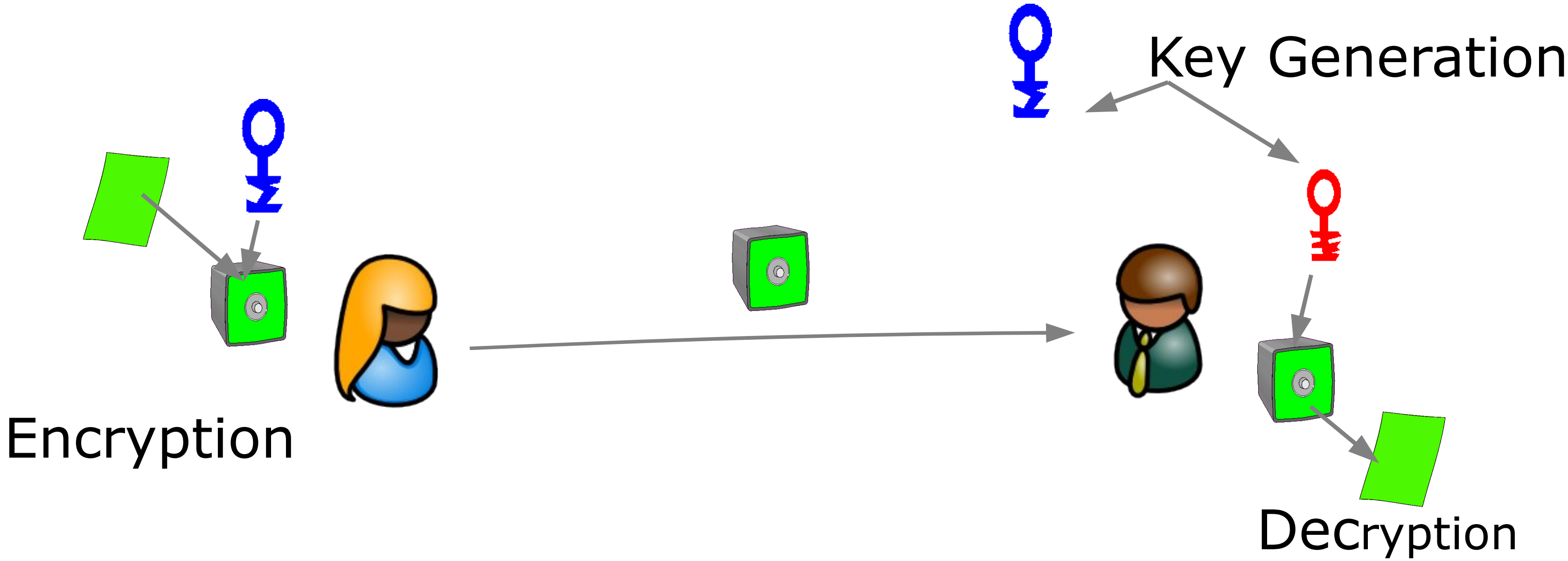


If car is broken: ID with insurance needs be retrieved

Can verifiably encrypt any certified attribute *(optional)*

TTP is off-line & can be distributed to lessen trust

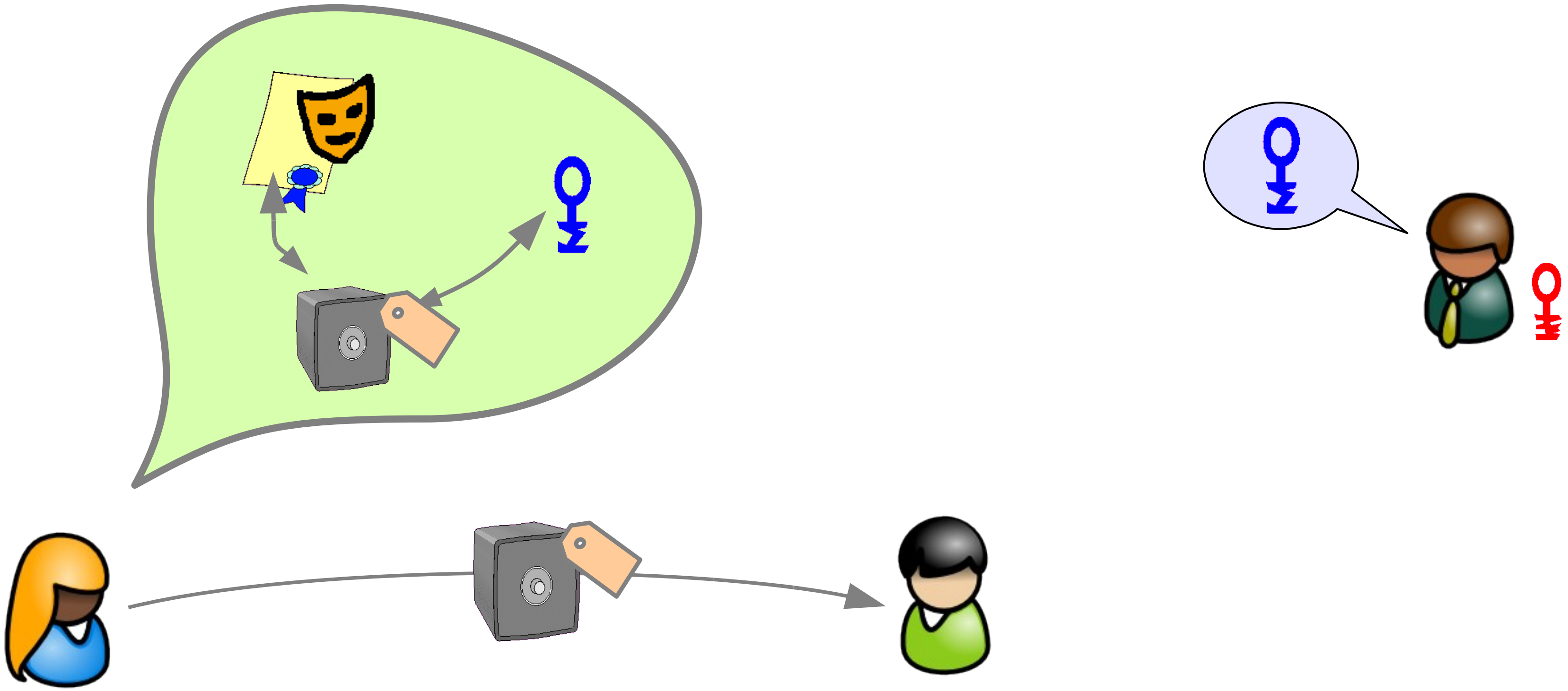
Public Key Encryption: algorithms



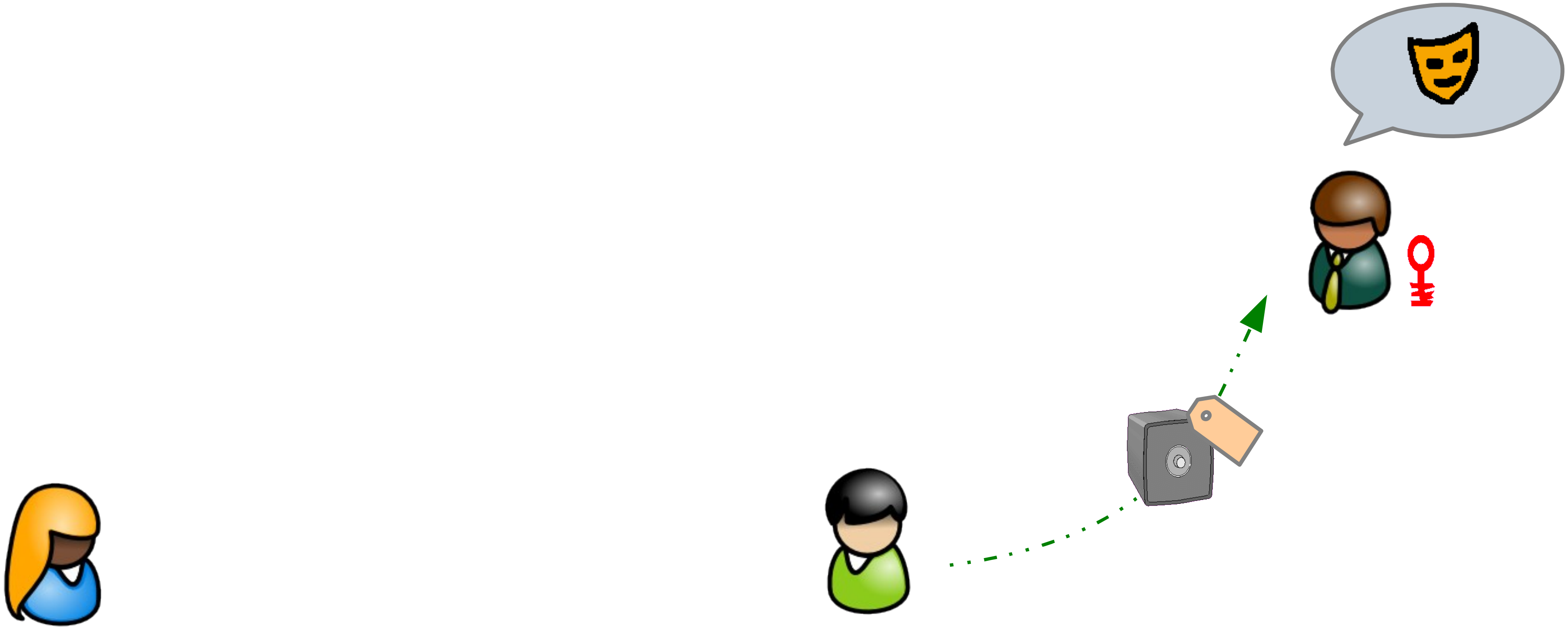
Security Definitions (far from trivial...)

- Semantic security: ciphertext does not leak if scheme is used once only.
- Adaptive security: .... if used continuously.

Verifiable Encryption with Label



Verifiable Encryption with Label



Verifiable Encryption

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

ElGamal Encryption

(for Pseudonyms & Tracing)

ElGamal Encryption scheme

Public Key: Group $G = \langle g \rangle$ of order q $y := g^x$

Secret Key: $x \in Zq$

Encryption of a message $m \in G$:

- Choose $r \in Zq$
- Compute ciphertext $(c1,c2)$ as: $c1 := g^r$, $c2 := y^r m$

Decryption of a ciphertext $(c1,c2)$:

- Compute $m' = c2\ c1^{-x}$ $(= y^r m g^{-rx} = y^r m y^{-r} = m\ )$

Semantically secure under Discrete Log assumption. *Cramer-Shoup encryption scheme is adaptive secure extension that should be used.*

Tracing: Encryption of a Certified Pseudonym

Public Key Of Tracer: Group $G = \langle g \rangle$ of order q $y := g^{x}$

Pseudonym with issuer: $P := g^{sk} h^{r}$

…by definition of credential by issuer $d = c^{e} a_1^{sk} \cdot a_2^{r} \cdot a_3^{m3} \cdot \ldots \cdot a_k^{mk}$ b^{s}

To make a traceable presentation of credential, user

- Choses rand. r and computes $c1 := g^{r'}, c2 := y^{r'} P \; (= y^{r'} g^{sk} h^{r})$
- Computes $c' = c\, b^{s'}$ mod n with random s'
- Sends $(c', (c1, c2))$ to verifier
- Computes SPK{($sk, r, m2, .., mk$) :

$$c1 := g^{r'},\; c2 := y^{r'} g^{sk} h^{r} \quad \wedge$$

$$d = c'^{e} a_1^{sk} \cdot a_2^{r} \cdot a_3^{m3} \cdot \ldots \cdot a_k^{mk}\, b^{s}\}(c1, c2, tr\_policy)$$

Excursus: Accountability

Making the User Accountable: Discussion

Scenario:

- User registers pseudonym with issuer
- Get credential from the issuer
- Presents credential to verifier with tracing (encryption of registered pseudonym)
- TTP traces a presentation proof and points to a user

Problems/attacks:

- How can one prove that it really was the user?
- Could the issuer just generate another pseudonym and credential and then blame the user?

Making the User Accountable: Solution

Assume: user has traditional government issued signing key and certificate

Then:

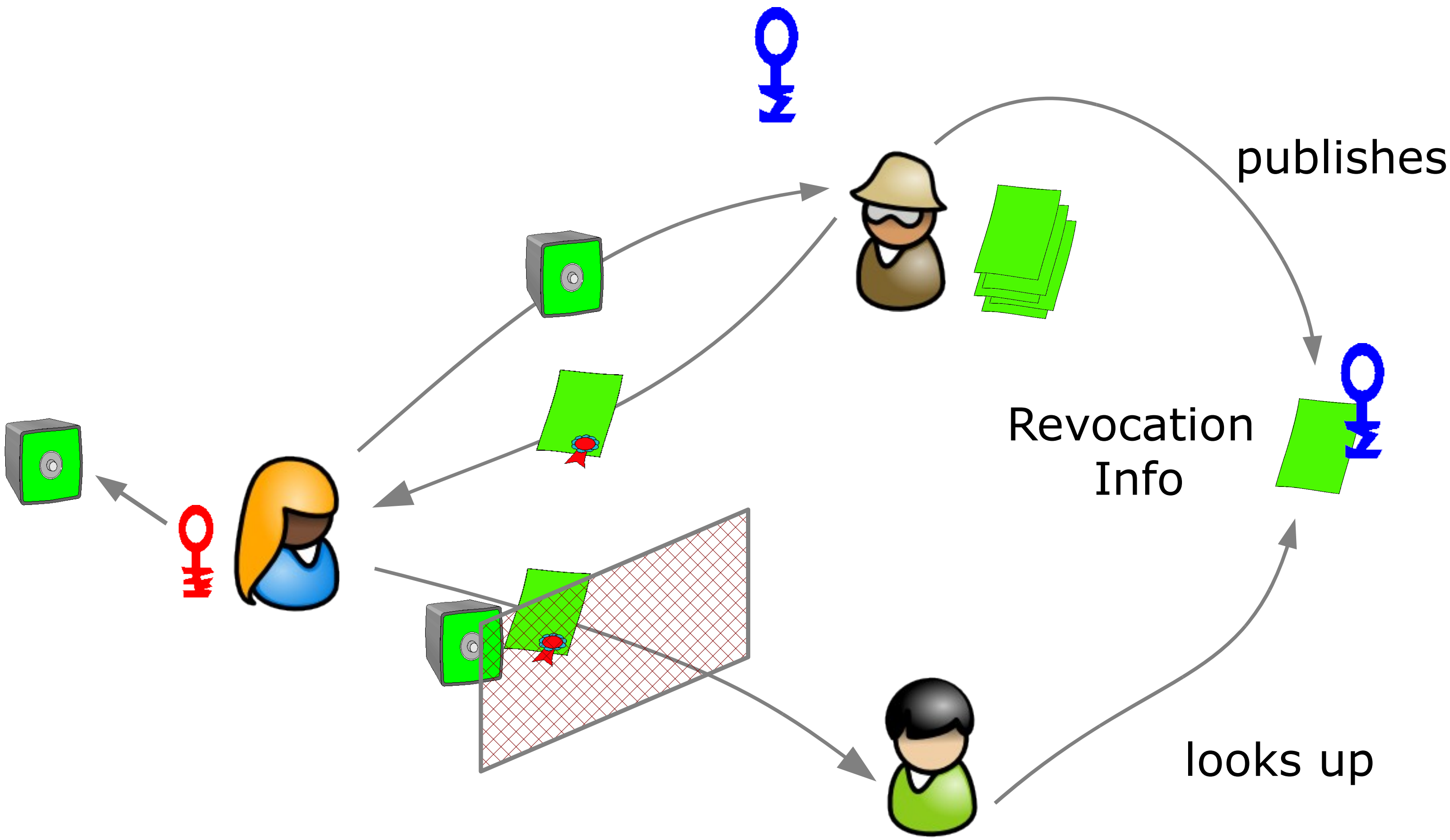
- User generates pseudonym P and *signs it* with gov't issued signing key, registers pseudonym and sign with issuer.
- Gets credential issued on pseudonym.
- Presents credential with tracing enabled (encryption of pseudonym) – *needs to be non-interactive presentation/signature.*
- Tracers claims this was the user holding pseudonym P ????

Convincing a judge:

- Verify users' non-interactive presentation proof
- Tracer needs to prove that encryption in presentation proof indeed contains P
- Issuer needs provide user's signature with gov't issued key on P

Revocation Methods

Revocable Credentials: Scenario



Alice should be able to convince verifier that her credential is not revoked (yet)!

Different Methods

CRL based methods
- traditional serial number & proof of non-membership
  - Best method uses signatures on pairs of succeeding revoked serial numbers
- Accumulator based solution
  - Presentation proof is efficient, but users need to update for each revoked credential

Short lived credentials
- Re-issuing (can always be done done, but requires interaction....)
- Publishing updates of credentials (define validity epochs), compute updates off-line

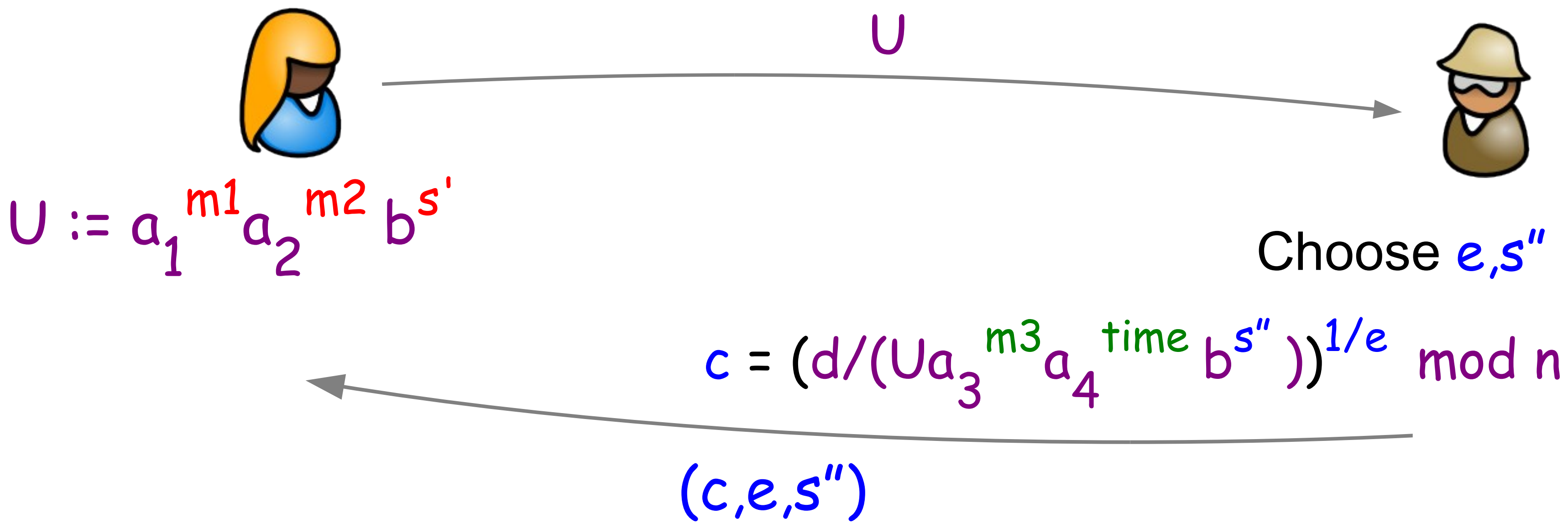
Verifier-Local “revocation”
- Essentially uses domain pseudonym and

Notes:
- Choice on method depend on use case and can also be combined
- All methods work for both U-Prove and idemix signatures - excepts off-line credential update does not work for U-Prove signature scheme (needs interaction)

Revocation: Zeroth Solution

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

- Recall issuing for identity mixer:



Revocation: Zeroth Solution

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

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

Choose en,sn"

$$cn = (d/(C a_3^{m3'} a_4^{time'} b^{sn''}))^{1/en} \bmod n$$

(cn,en,sn")

Revocable Credentials: First Solution

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

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

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

$$(\#1, \ldots, \#k)$$

- Alice proves that her # is (or is not) on the list.
  - Compute $Uj = g^{\#j}$ for $\#j$ in $(\#1, \ldots, \#k)$
  - Prove $PK\{(\varepsilon, \mu, \rho, \sigma) : \quad d = c'^{\varepsilon} a_1^{\rho} a_2^{\mu} b^{\sigma} \pmod{n} \wedge$

$$(U1 = g^{\mu} \vee \cdot \quad \cdot \quad \cdot \vee \; Uk = g^{\mu}) \}$$

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

A better implementation of this idea where the issuer signs pairs of serial numbers (i.e., sig(#i,#i+1)) and have the user prove knowledge of sig(#i,#i+1) such that #i < # < #i+1 (c.f.

Revocable Credentials: Second Solution

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

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

- Publish list of all *invalid* #i's.

$$(\#1, \ldots, \#k)$$

- Alice proves that her ui is on the list.
  - Choose random h and compute $U = h^{\#i}$
  - Prove $PK\{(\varepsilon, \mu, \rho, \sigma) : \quad d = c'^{\varepsilon} a_1^{\rho} a_2^{\mu} b^{\sigma} \pmod{n} \ \wedge\ U = h^{\mu} \}$
  - Verifier checks whether $U = h^{\#j}$ for all #j 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 #i's public?

Revocable Credentials: Second Solution

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

- Can pre-compute list $U_i = h^{x_i}$
- -> single table lookup

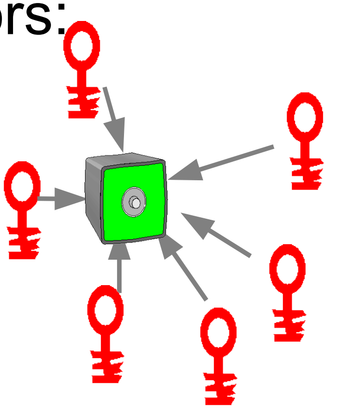
- BUT: if user comes again, verifier can link!!!
- ALSO: verifier could not change h at all! or use the same as other verifiers!
  - one way out $h = H(\text{verifier, date})$, so user can check correctness.
  - date could be the time up to seconds and the verifier could just store all the lists, i.e.,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

Note: This is the method implemented in TPM's Direct Anonymous Attestation, where #i is a secret only known to the user. Thus credentials

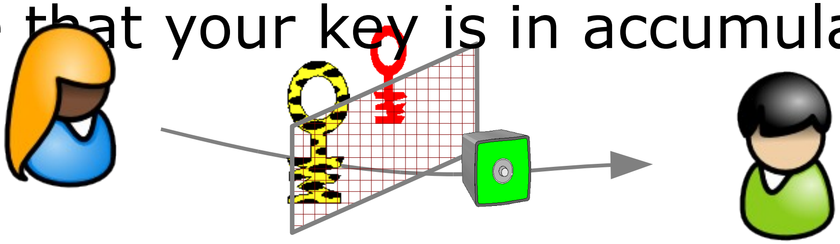
Revocable Credentials: Third Solution

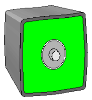
Using so-called cryptographic accumulators:

- Accumulate:

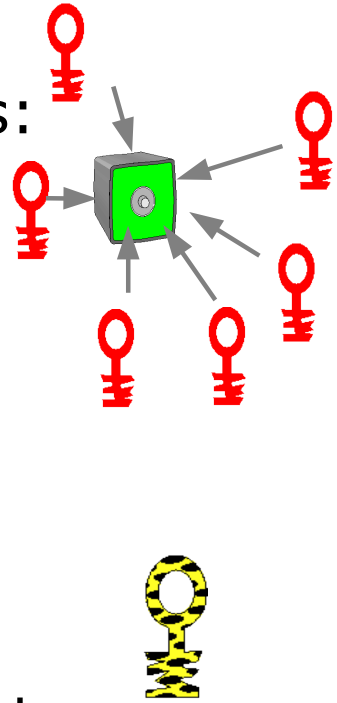


- Prove that your key is in accumulator: requires witness





Revocable Credentials: Third Solution



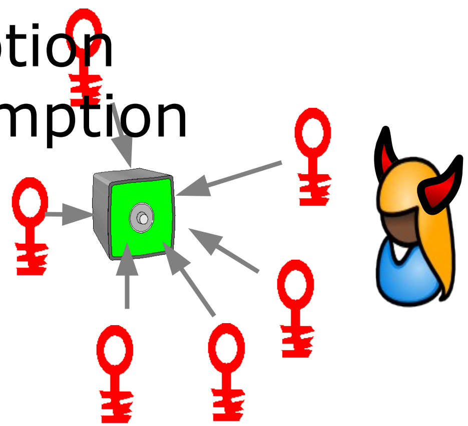
Using so-called cryptographic accumulators:

- Key setup: RSA modulus n, seed v
- Accumulate:
  - values are primes ei
  - accumulator value: $z = v^{\Pi\, ei} \bmod n$
  - 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 $z = x^{e} \bmod n$

Revocable Credentials: Third Solution

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

Revocable Credentials: Third Solution

- Prove that your key is in accumulator:
  - choose random s and g and compute $U1 = x\,h^{s}$ (where h is a publicly known value such that it is assured that x lies in $\langle h \rangle$) and compute $U2 = g^{s}$ and reveal $U1$, $U2$, g
  - Run proof-protocol with verifier

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

Revocable Credentials: Third Solution

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

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

Revocation: Third Solution

Dynamic Accumulator

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

Revocation: Third Solution

Dynamic Accumulator

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

Revocation: Third Solution (improved)

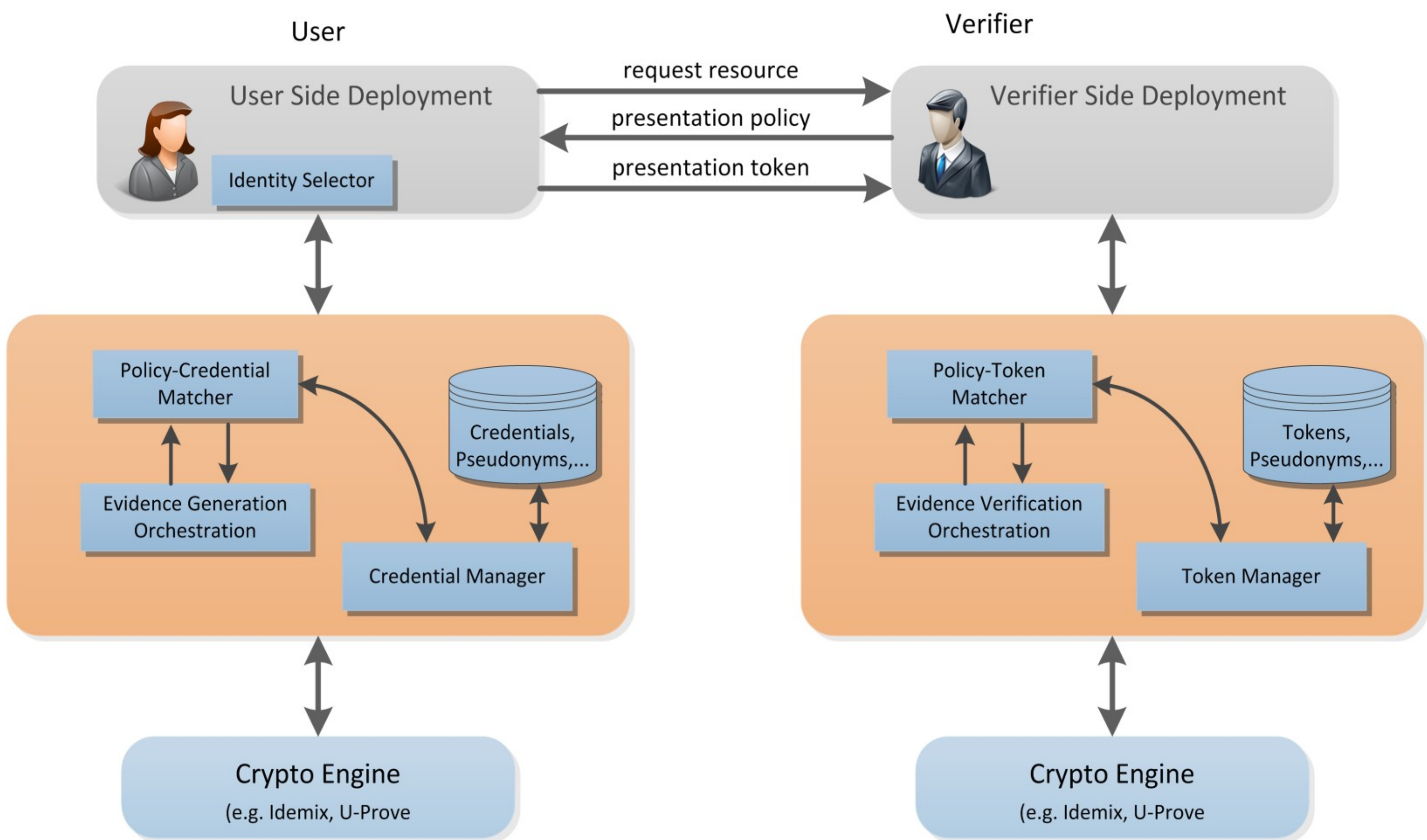
Dynamic Accumulator: in case the issuer knows the factorization of n

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

- Witnesses need to be recomputed upon revocation only!

Architecture and Policies

User – Verifier interaction: an architectural view [abc4trust.eu]



Concepts of ABC technologies to be defined

Technology-agnostic XML schemas for "external" artefacts, including:

Issuance

- Pseudonyms
- Issuer parameters
- Credential specification
- Issuance policies
- Issuance token

Using credentials

- Verifier parameters
- (Pseudonyms)
- Presentation policies
- Presentation tokens

Revocation, Issuer & Verifier driven

- Revocation authority parameters
- cf. Presentation token

Inspection

- Inspector parameter
- cf. Presentation token
  - Inspection grounds

Credential specification

E.g., School credentials

```xml
<CredentialSpecification Version="1.0" KeyBinding="true" Revocable="true">

    <SpecificationUID>http://abc4trust.eu/wp6/credspec/credSchool</SpecificationUID>

    <AttributeDescriptions MaxLength="32">
        <AttributeDescription Type="http://abc4trust.eu/wp6/credspec/credSchool/firstName"
         DataType="xs:string" Encoding="abc:sha256"/>
        <AttributeDescription Type="http://abc4trust.eu/wp6/credspec/credSchool/lastName"
        DataType="xs:string" Encoding="abc:sha256"/>
        <AttributeDescription Type="http://abc4trust.eu/wp6/credspec/credSchool/civicNr"
        DataType="xs:integer" Encoding="abc:plain"/>
        <AttributeDescription Type="http://abc4trust.eu/wp6/credspec/credSchool/gender"
        DataType="xs:boolean" Encoding="abc:zero-one"/>
        <AttributeDescription Type="http://abc4trust.eu/wp6/credspec/credSchool/school"
        DataType="xs:string" Encoding="xenc:sha256"/>
    </AttributeDescriptions>

</CredentialSpecification>
```

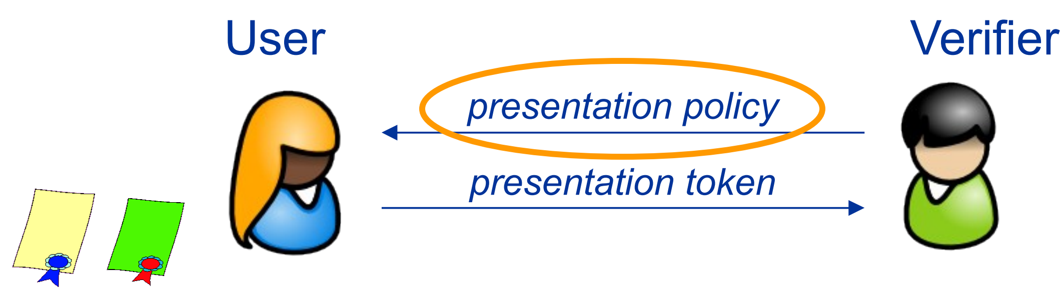
Issuer parameters

```
<IssuerParameters>

    <ParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/school</ParametersUID>
    <AlgorithmID>urn:com:microsoft:uprove</AlgorithmID>
    <SystemParameters>...</SystemParameters>
    <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credSchool</CredentialSpecUID>
    <HashAlgorithm>http://www.w3.org/2001/04/xmlenc#sha256</HashAlgorithm>
    <CryptoParams>...</CryptoParams>
    <KeyBindingInfo>...</KeyBindingInfo>
    <RevocationParametersUID>http://abc4trust.eu/wp6/soderhamn/RevParams/school</RevocationParametersUID>

</IssuerParameters>
```

Presentation policy

“reveal civic number from school credential”

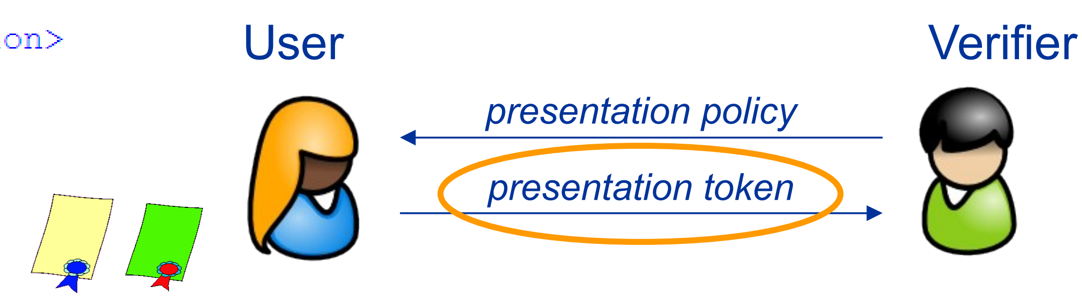
```xml
<PresentationPolicyAlternatives>
    <PresentationPolicy PolicyUID="revealCivicNr">
        <Message>
            <Nonce>bkQydHBQWDR4TUZzbXJKYUphdVM=</Nonce>
        </Message>
        <Credential Alias="schoolcred">
            <CredentialSpecAlternatives>
                <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credSchool</CredentialSpecUID>
            </CredentialSpecAlternatives>
            <IssuerAlternatives>
                <IssuerParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/school</IssuerParametersUID>
            </IssuerAlternatives>
            <DisclosedAttribute AttributeType="http://abc4trust.eu/wp6/credspec/credSchool/civicNr"/>
        </Credential>
    </PresentationPolicy>
</PresentationPolicyAlternatives>
```



Presentation token

“reveal civic number from school credential”

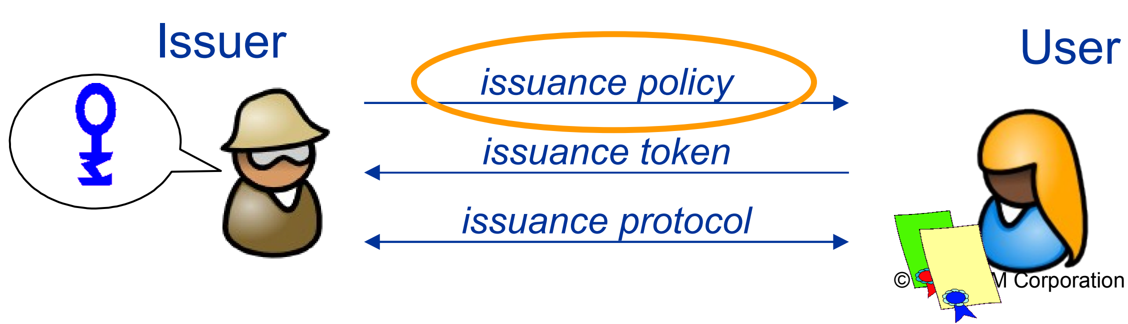
```
<PresentationToken>
    <PresentationTokenDescription PolicyUID="revealCivicNr">
        <Message>
            <Nonce>bkQydHBQWDR4TUZzbXJKYUphdVM=</Nonce>
        </Message>
        <Credential Alias="schoolcred">
            <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credSchool
            </CredentialSpecUID>
            <IssuerParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/school
            </IssuerParametersUID>
            <DisclosedAttribute AttributeType=
            "http://abc4trust.eu/wp6/credspec/credSchool/civicNr">
                <AttributeValue>19980225l234</AttributeValue>
            </DisclosedAttribute>
        </Credential>
    </PresentationTokenDescription>
    <CryptoEvidence>
        ...
    </CryptoEvidence>
</PresentationToken>
```



Issuance policy

Carry over key from school credential to course credential

```xml
<IssuancePolicy>
    <PresentationPolicy PolicyUID="revealCivicNr">
        <Credential Alias="schoolcred">
            <CredentialSpecAlternatives>
                <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credSchool
                </CredentialSpecUID>
            </CredentialSpecAlternatives>
            <IssuerAlternatives>
                <IssuerParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/school
                </IssuerParametersUID>
            </IssuerAlternatives>
        </Credential>
    </PresentationPolicy>
    <CredentialTemplate SameKeyBindingAs="schoolcred">
        <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credCourse
        </CredentialSpecUID>
        <IssuerParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/course
        </IssuerParametersUID>
    </CredentialTemplate>
</IssuancePolicy>
```



Presentation policy

- Boys older than 12 taking English
- Civic number recoverable by school inspector

```xml
<PresentationPolicyAlternatives>
    <PresentationPolicy PolicyUID="existing">
        <Message>
            <Nonce>bkQydHBQWDR4TUZzbXJKYUphdVM=</Nonce>
        </Message>
        <Pseudonym Scope="http://soderhamn.se/highschool/discuss" Established="true"/>
    </PresentationPolicy>

    <PresentationPolicy PolicyUID="new">
        <Message>
            <Nonce>bkQydHBQWDR4TUZzbXJKYUphdVM=</Nonce>
        </Message>
        <Pseudonym Scope="http://soderhamn.se/highschool/discuss" Alias="nym"/>
        <Credential Alias="school" SameKeyBindingAs="nym">
            <CredentialSpecAlternatives>
                <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credSchool</CredentialSpecUID>
            </CredentialSpecAlternatives>
            <IssuerAlternatives>
                <IssuerParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/school</IssuerParametersUID>
            </IssuerAlternatives>
            <DisclosedAttribute AttributeType="http://abc4trust.eu/wp6/credspec/credSchool/civicNr">
                <InspectorPublicKeyUID>http://abc4trust.eu/wp6/soderhamn/SchoolInspector</InspectorPublicKeyUID>
                <InspectionGrounds>Concrete safety threat.</InspectionGrounds>
            </DisclosedAttribute>
        </Credential>
```

Presentation policy (cont.)

– Boys older than 12 taking English

– Civic number recoverable by school inspector

```
    <Credential Alias="subject" SameKeyBindingAs="school">
        <CredentialSpecAlternatives>
            <CredentialSpecUID>http://abc4trust.eu/wp6/credspec/credSubject</CredentialSpecUID>
        </CredentialSpecAlternatives>
        <IssuerAlternatives>
            <IssuerParametersUID>http://abc4trust.eu/wp6/soderhamn/IssParams/subject</IssuerParametersUID>
        </IssuerAlternatives>
    </Credential>
    <AttributePredicate Function="urn:oasis:names:tc:xacml:1.0:function:boolean-equal">
        <Attribute CredentialAlias="school" AttributeType=
        "http://abc4trust.eu/wp6/credspec/credSchool/gender"/>
        <ConstantValue>false</ConstantValue>
    </AttributePredicate>
    <AttributePredicate Function="urn:oasis:names:tc:xacml:1.0:function:integer-less-than">
        <Attribute CredentialAlias="school" AttributeType=
        "http://abc4trust.eu/wp6/credspec/credSchool/civicNr"/>
        <ConstantValue>200002139999</ConstantValue>
    </AttributePredicate>
    <AttributePredicate Function="urn:oasis:names:tc:xacml:1.0:function:string-equal">
        <Attribute CredentialAlias="subject" AttributeType=
        "http://abc4trust.eu/wp6/credspec/credSubject/subject"/>
        <ConstantValue>English</ConstantValue>
    </AttributePredicate>
  </PresentationPolicy>
</PresentationPolicyAlternatives>
```

Overview ABC4Trust

State of the Art & Project Goals

Attribute based credentials require crypto algorithms different from those currently used:

- RSA to sign credentials/certificates as done today would not work ...

U-Prove and Identity Mixer provide such crypto algorithms.

Attribute based authentication is a paradigm shift in authentication:

Attributes instead of name-based identifiers

- Teenage chat room: „Between 12 and 15" instead of name-based identifier

Paradigm shift and interoperability in trustworthy infrastructures require:

- Abstraction and unification of different crypto algorithms
- Interaction flows & Architecture
- Policies (Claims language)
- Data formats
- Reference implementation
- Validation by real world pilots in the eID space

Work Packages

1) Architectures & Components
   - Modular Decomposition
   - Common Formats and APIs
   - Protocol Definitions
2) Comparison
   - Comparison of Different Implementations of Components
   - Security Proofs & Perturbation Analysis
3) Reference Implementation
   - At least two different ones (guess what)
4) Application Requirements
   - Common Base & Infrastructure for Prototypes
5) Community Interactions Among Pupils
   - Swedish Community
6) Course Rating by Certified Students
   - Greek Ministry of Education
7) Dissemination
8) Management

References

- www.abc4trust.eu
- https://github.com/p2abcengine/p2abcengine
- Jan Camenisch, Anna Lysyanskaya: *Efficient non-transferable anonymous multishow credential system with optional anonymity revocation*. In EUROCRYPT 2001, vol. 2045 of LNCS, pp. 93–118. Springer Verlag, 2001
- David Chaum. Security without identification: *Transaction systems to make big brother obsolete*. Communications of the ACM, 28(10):1030–1044, Oct. 1985.
- 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