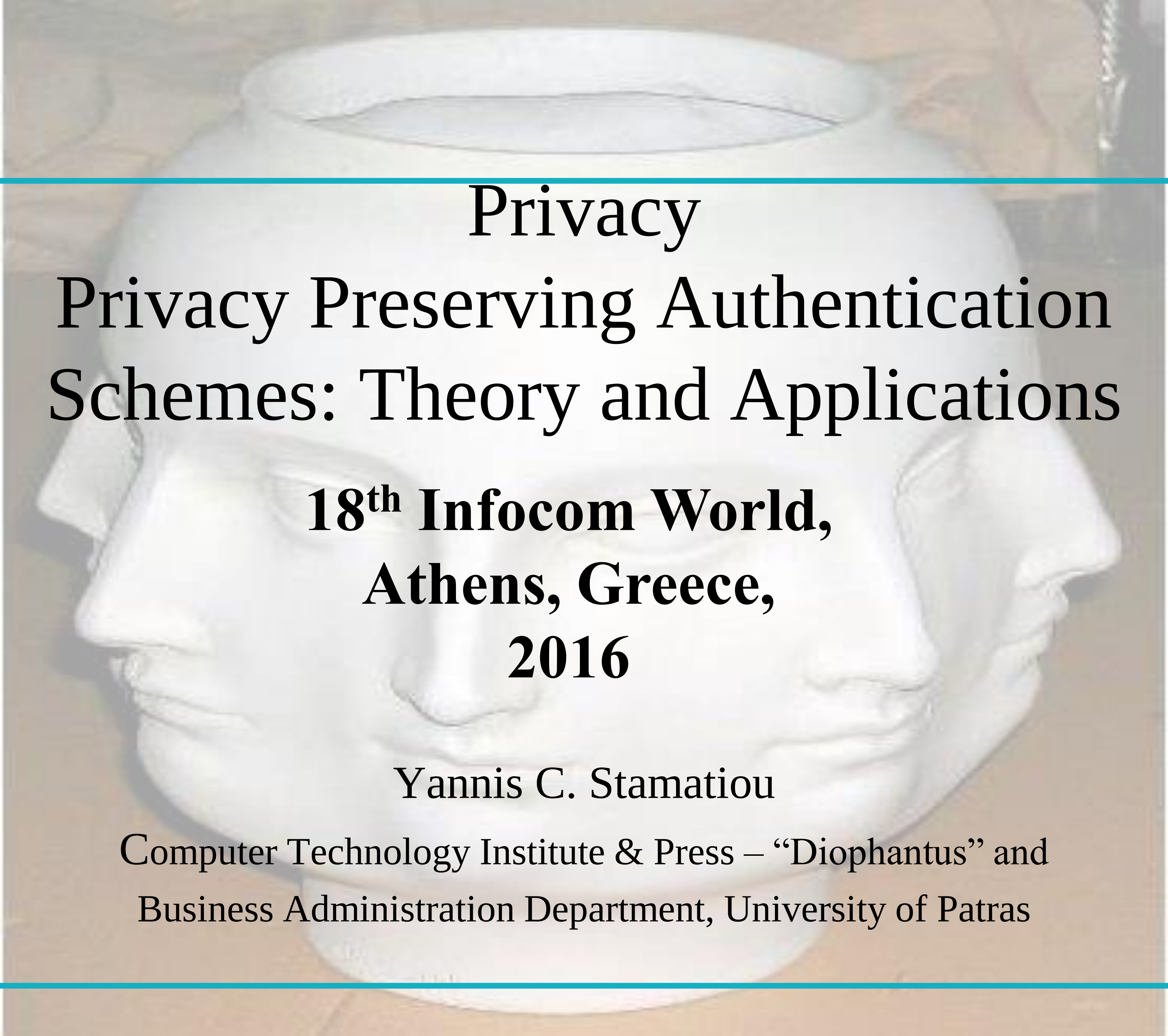


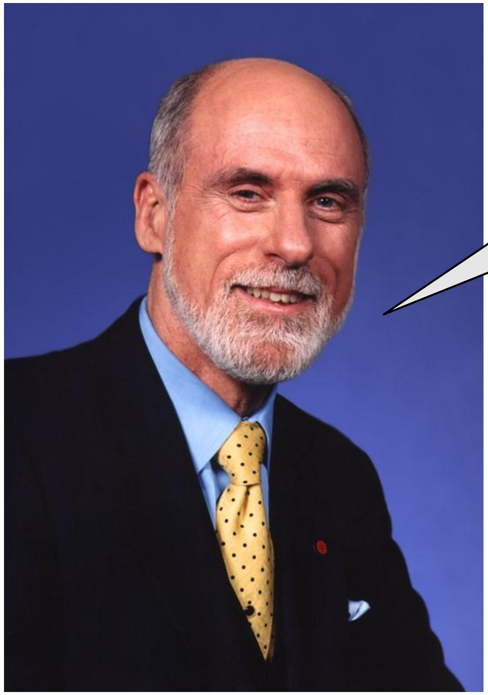
Privacy
Privacy Preserving Authentication Schemes: Theory and Applications

18th Infocom World,
Athens, Greece,
2016

Yannis C. Stamatiou

Computer Technology Institute & Press – “Diophantus” and
Business Administration Department, University of Patras

“The Internet will be everywhere, from every mote to interstellar communication”

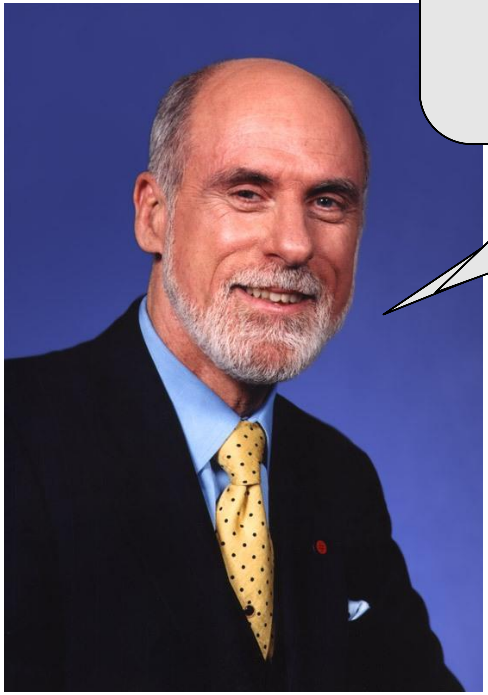


Vint Cerf (one of the fathers of the Internet!)

“The Internet will be everywhere, from every

cation”

“The Internet needs to have a secure identity layer”



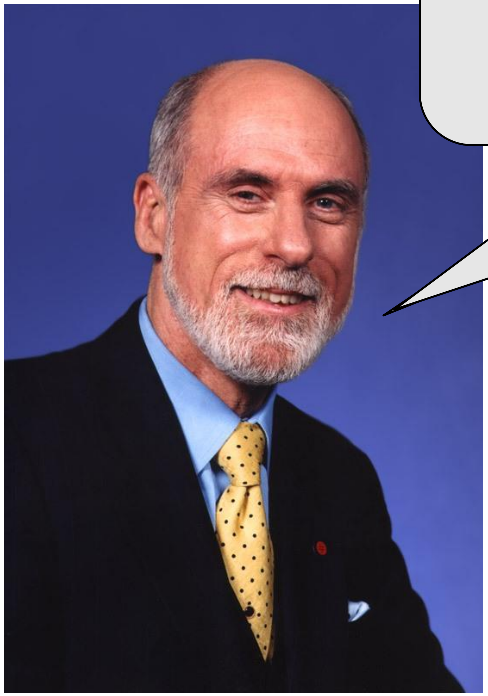
Vint Cerf (one of the fathers of the Internet!)

“The Internet will be everywhere, from every

cation”

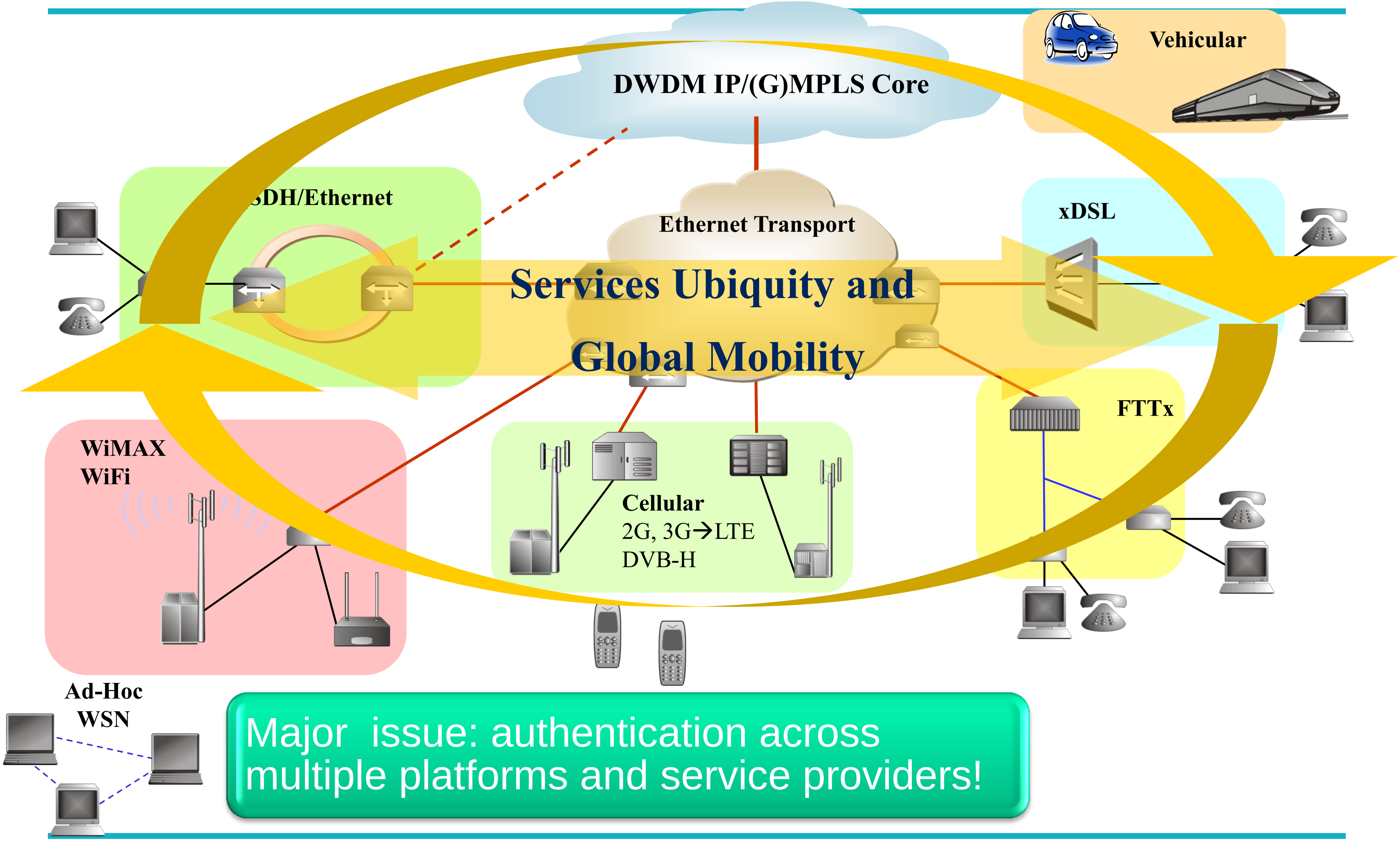
“The Internet needs to have a secure

“We need both: sometimes we wanna be anonymous, sometimes we need to be identified”



Vint Cerf (one of the fathers of the Internet!)

Multi-Network Services



Major issue: authentication across multiple platforms and service providers!

Identity landscape

- More and more business/government services are migrated online, since this
  - improves convenience (both for users and the service provider)
  - reduces costs, and
  - helps publicity.

- High-value transactions require high-level of identity assurance
  - Usernames/passwords are ubiquitous, but provide low-security (NIST's LoA – Levels of Assurance)
  - Conventional "enterprise" solutions (e.g., Kerberos, PKI) don't scale or are not flexible enough for an internet-wide system
  - How can you show some ID online, just like in real life?

Identity federation

- In information technology, **federated identity** has two general meanings:
  - The virtual reunion, or *assembled identity*, of a person's user information (or principal), stored across multiple distinct identity management systems. Data are joined together by use of the common token, usually the user name.
  - **[The meaning we will discuss]** A user's authentication process across multiple IT systems or even organizations.
- For example, a traveler is a flight passenger and a hotel guest.

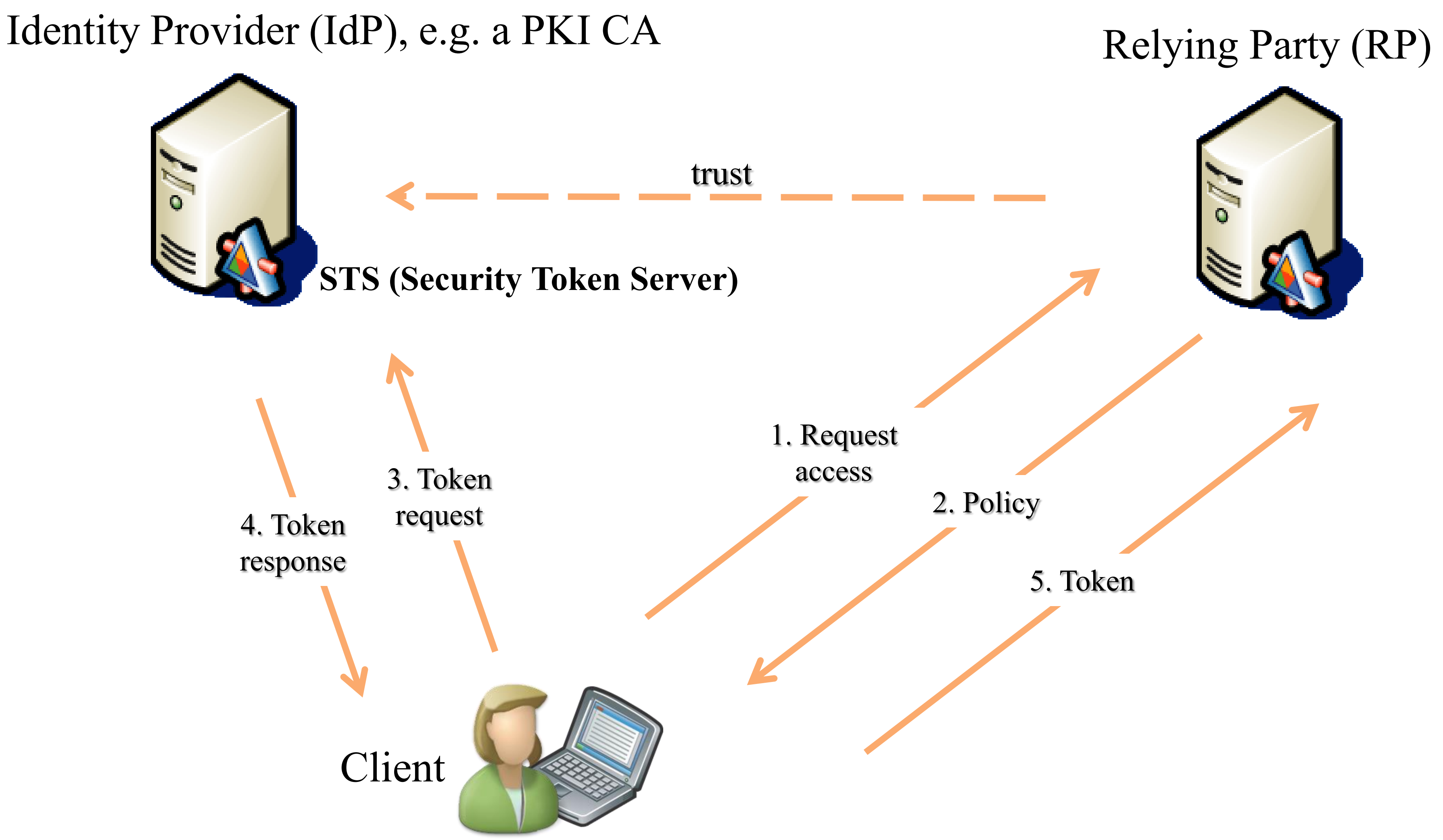
  If the airline and the hotel use a federated identity management system, they have a contracted mutual trust in each other's authentication of the user.

  The traveler could identify him/herself once as a customer for booking the flight and this identity can be carried over to be used for the reservation of a hotel room.

Identity federation

- Most popular proposed framework for identity management
  - Very flexible
  - Easy to deploy
  - Many protocols: WS-Federation/Trust, SAML, Information Cards, OpenID, OAuth, …
- But many challenges exist:
  - Security
  - Privacy
  - Scalability

Federated architecture



Challenge #1: Security

- Compromise IdP credential, access all RPs
  - Phishing problem
- Strong authentication to IdP is possible, but authentication to RP is weaker
  - Issued tokens are software only (token hijacking attacks, transferability)
- IdP is all powerful
  - IdP (insider, malicious code) can surreptitiously act on the users' behalf
  - Selectively deny access

Challenge #2: Privacy

- IdP can profile users’ activities
- Even if IdP doesn’t learn the visited RP, profiling is possible by colluding parties (or insiders)
  - Timing correlation
  - Unique correlation handles (e.g., digital signatures, serial numbers, etc.)

Challenge #3: Scalability

- All tokens are retrieved on-demand
  - IdP must be available 24/7

- IdP is a central point of failure
  - Nice target for denial of service attack

- IdP is a bottleneck for every user access

PETs Can Help! - A More Structured Approach

- PETs: Privacy Enhancing Technologies based on Privacy Enhancing Cryptography
- 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, ...

ABC Technology

- ABCs: Attribute Based Credentials

- Crypto technology combining the security of PKI with the flexibility of federation, providing *privacy-by-design*

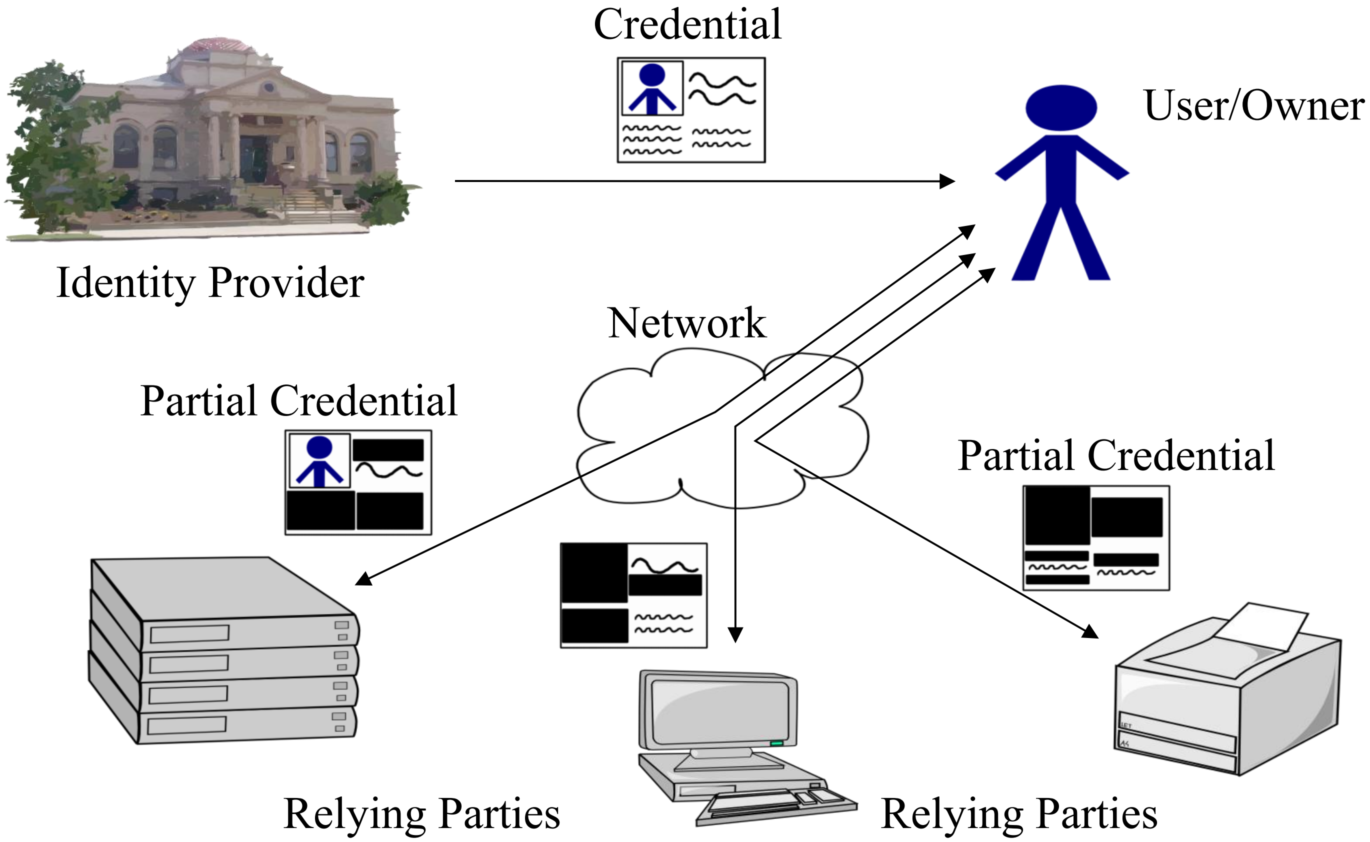
- Can be used to build various types of electronic credentials and entitlement documents

- Has unique security, privacy, and efficiency benefits over “conventional” crypto tokens (X.509 certificates, SAML assertions, Kerberos tickets)

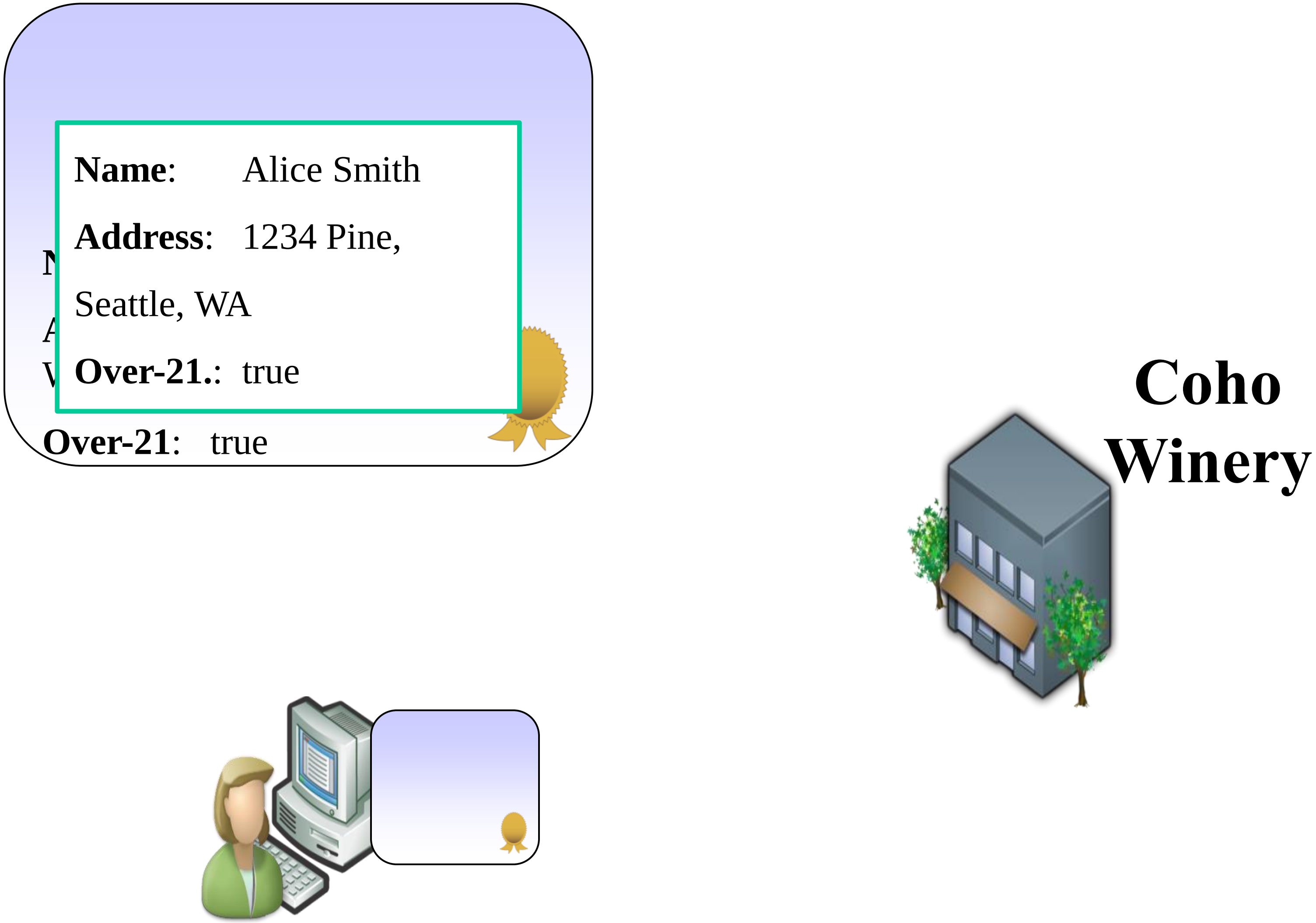
What’s new? Minimal disclosure!

- ABC tokens cannot be combined and lead to the revelation of identity
  - Token issuance and presentation are unlinkable
  - Think “coins” (cannot be distinguished) vs. “bills” (have a serial number!)
- Users can disclose a *subset* of the encoded claims
  - To respond to unanticipated requests of RPs
  - Without invalidating the token integrity

Minimal Disclosure Credentials: Use



An example of minimal disclosure:



N

A

V

Over-21: true

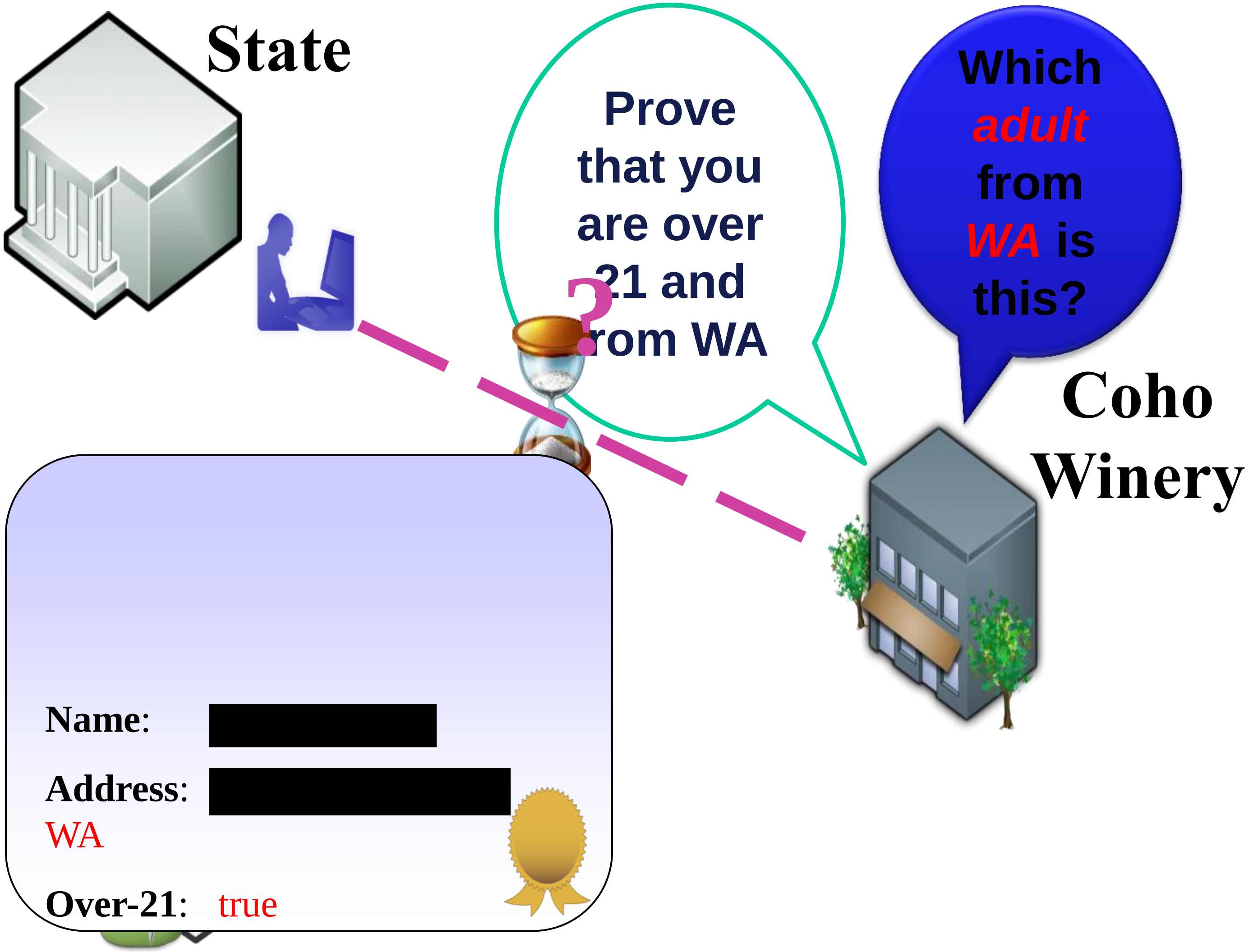
Name: Alice Smith

Address: 1234 Pine,

Seattle, WA

Over-21.: true

Minimal disclosure illustrated



State

Prove that you are over 21 and from WA

?

Which *adult* from *WA* is this?

Coho Winery

Name: [redacted]

Address: [redacted]

WA

Over-21: true

Underlying crypto

- Based on the Brands protocols
  - 30+ papers (from ‘93 onward)
  - Evolution of PKI
  - MIT Press book, foreword by Ron Rivest

- Issuance uses a “restrictive blind signature”
  - Issuer knows the attributes, but never sees the resulting public key and signature on tokens

- Presentation uses a ***proof of knowledge***
  - Prove a secret without leaking any info about it
  - Generalization of the Schnorr protocol

Zero-knowledge Proofs

- Interactive protocols between two players, Prover and Verifier, in which the prover proves to the verifier, with high probability, that some statement is true.
- Does not leak any information besides the veracity of this statement.
- In the case of honest verifier ZKP, we can modify the protocol to non-interactive.

How does a Zero Knowledge Proof work?

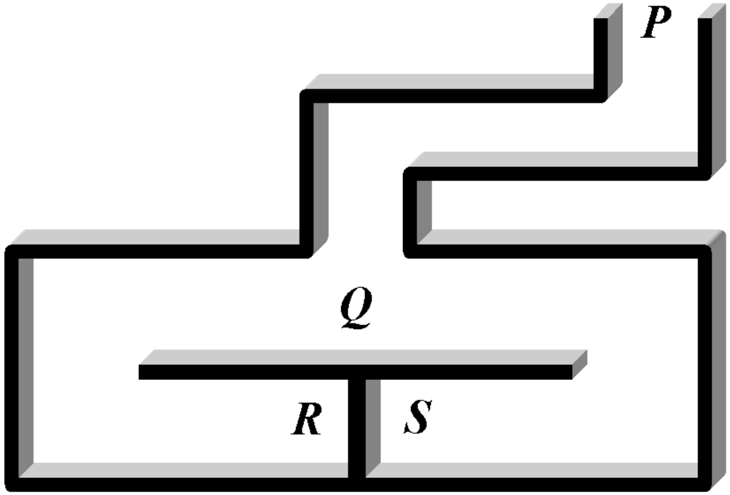


Image from RSA Labs [1]
http://www.rsasecurity.com/rsalabs/node.asp?id=2178

- Classic Example:
  - Ali Baba’s Cave
- Alice wants to prove to bob that she knows how to open the secret door between R and S.
  - Bob goes to P
  - Alice goes to R or S
  - Bob goes to Q and tells Alice to come from one side or the other of the cave
  - If Alice knows the secret, she can appear from the correct side of the cave every time
- Bob repeats as many times as he wants, until he is convinced that Alice really knows how to open the secret door!

The general setting

- Prover (P) tries to prove some fact to a verifier
- Verifier (V) either accepts or rejects the prover's proof
- To prove is to convince the verifier of some assertion
  - Prove that you know a secret value *s*
- Each party in the protocol does the following:
  1. receive a message from the other party
  2. perform a private computation
  3. send a message to the other party
- Repeats t number of rounds

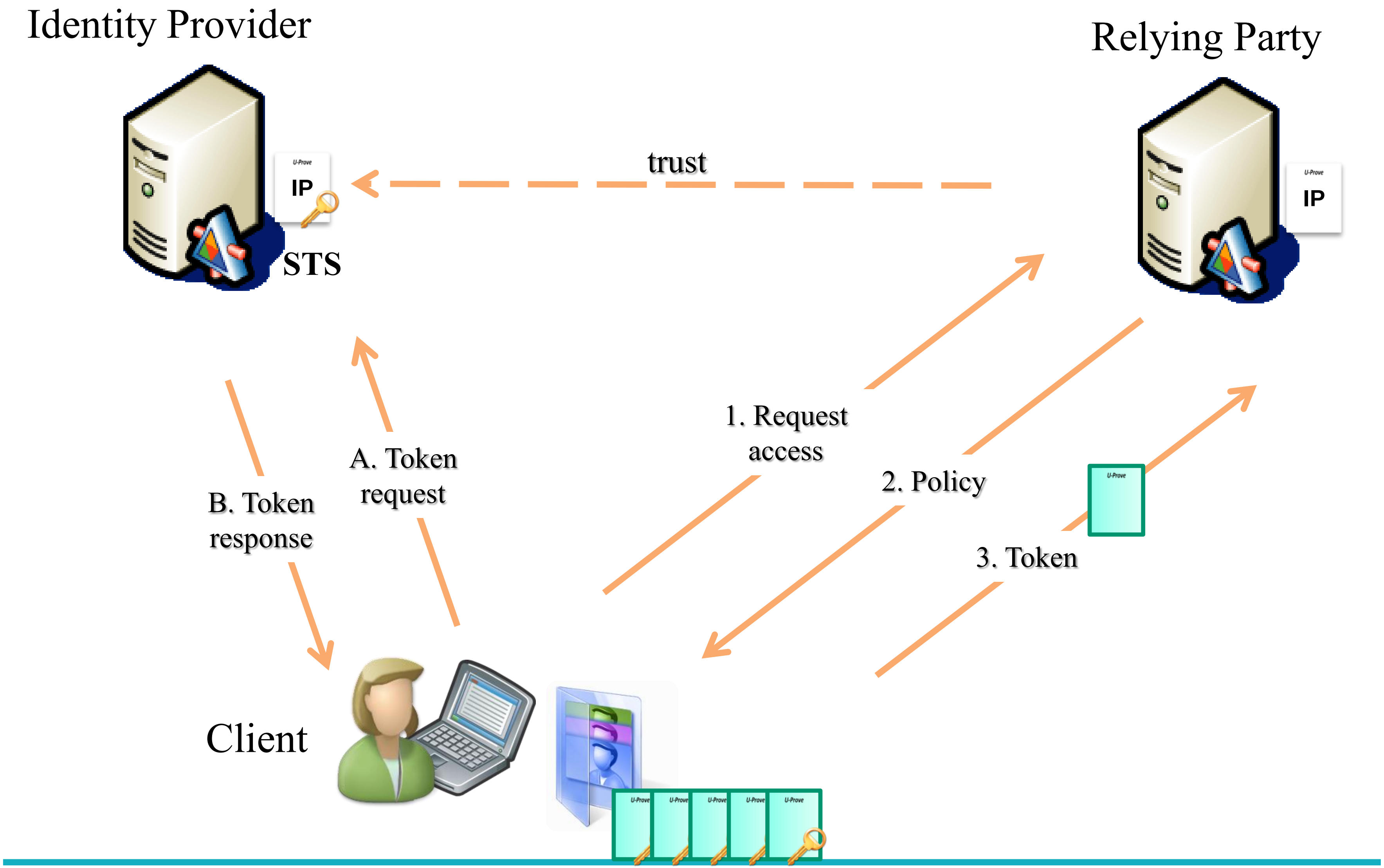
An more complex example:

- Let g_1, g_2 generators of Z_q^*.
- The Prover claims that $log_{g1}v = log_{g2}w$ (=x) for publicly known v, w, g_1, g_2.
  - P chooses random $z \in [1..q]$ and sends $a=g_1^z$, $b=g_2^z$.
  - V selects random $c \in [1..q]$ and sends it.
  - P sends $r = (z+cx)$.
  - V verifies that $g_1^r=av^c$ and $g_2^r=bw^c$
- Can be turned into non-interactive
  - $C = Hash(a,b,v,w)$.

Another example (from eVoting): Proof of encryption of a valid value

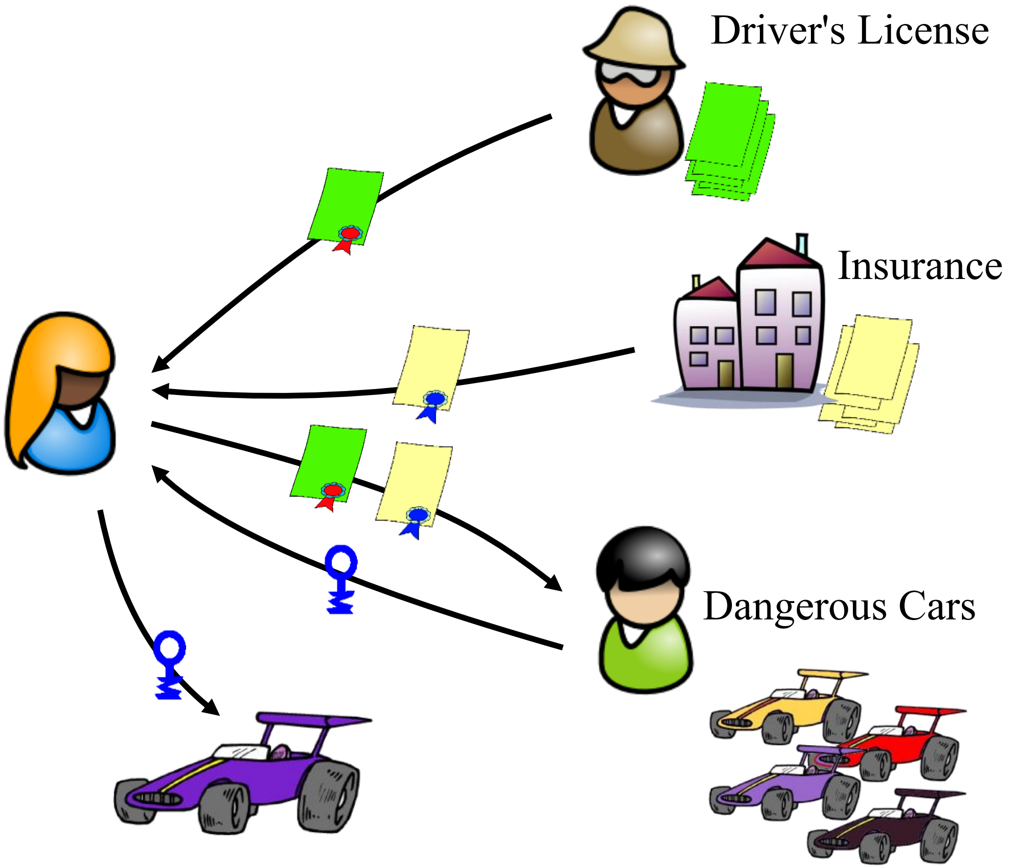
Voter	**V = -1**	**V = 1**
	Chooses random ω_1,ω_2, d_2, r_2, α	Chooses ω_1,ω_2, d_1, r_1, α
	Computes $B = g^{\alpha} \cdot h^{v}$	
	Computes a_2= $\left(B/h\right)^{-d_2} \cdot g^{r_2}$	Computes a_1= $\left(B \cdot h\right)^{-d_1} \cdot g^{r_1}$
	Computes a_1= g^{ω_1}	Computes a_2= g^{ω_2}
	Sends (B, a_1, a_2) to EA	
EA	Chooses random S and sends it to voter	
Voter	Computes d_2+d_1 = S	
	Computes $r_1 = \alpha d_1+\omega$	Computes $r_2 = \alpha d_2+\omega$
	Sends (r_1, r_2, d_1, d_2) to EA	
EA	Checks that d_1+d_2 = S	
	Checks that $(B \cdot h)^{d_1} \cdot a_1 = g^{r_1}$ and $(B/h)^{d_2} \cdot a_2 = g^{r_2}$	

Federation + ABCs



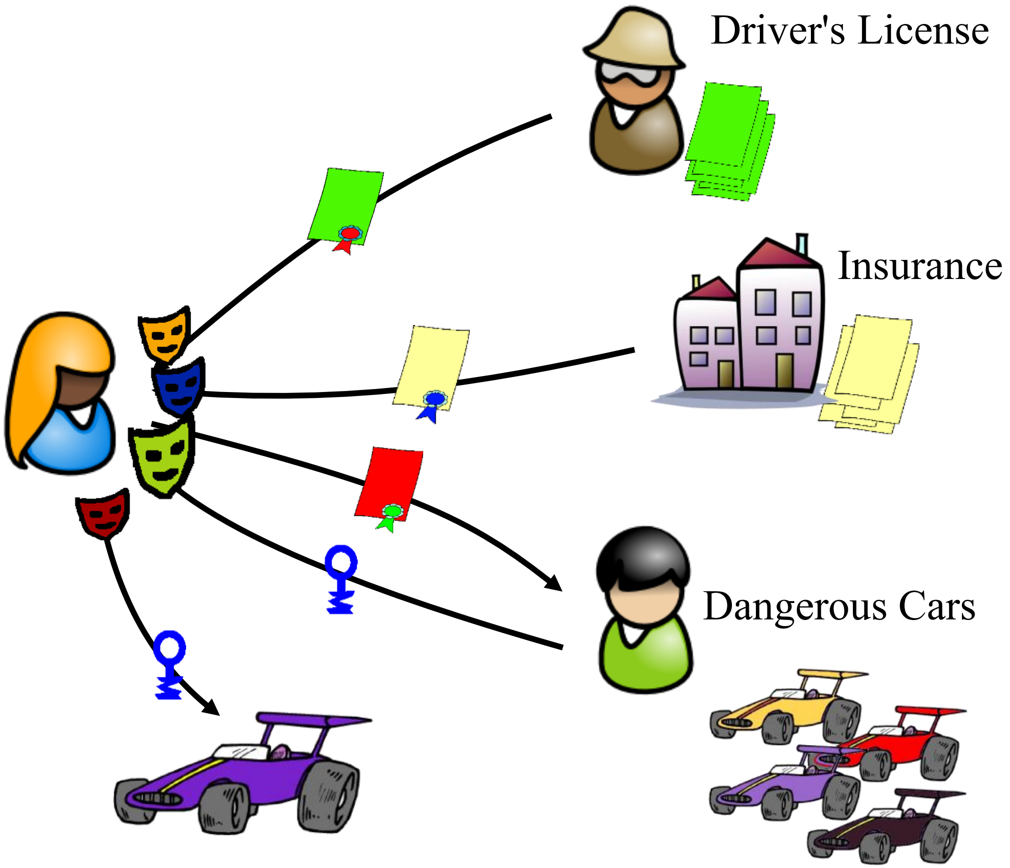
Digital Credentials

... or transmitting certified information



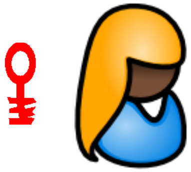
Private Digital Credentials (ABCs)

[Chaum, Damgaard, Brands,....]



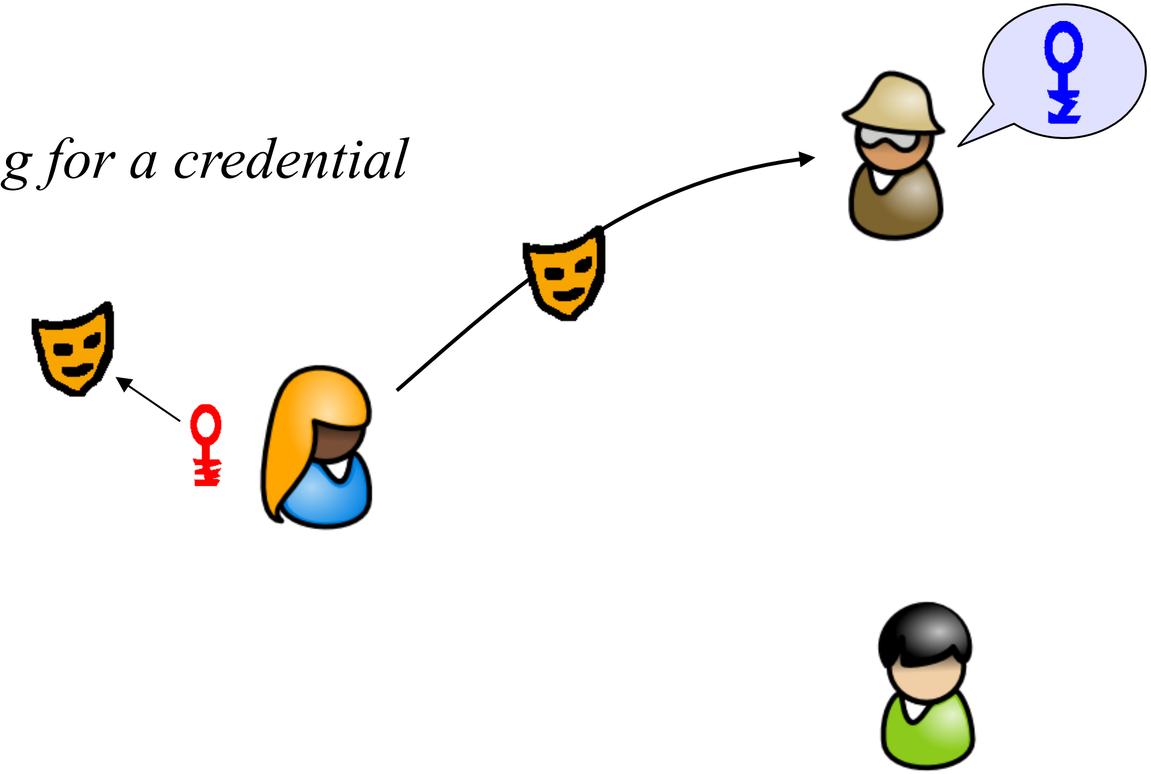
State of the Art: How to Build Them

In the beginning...

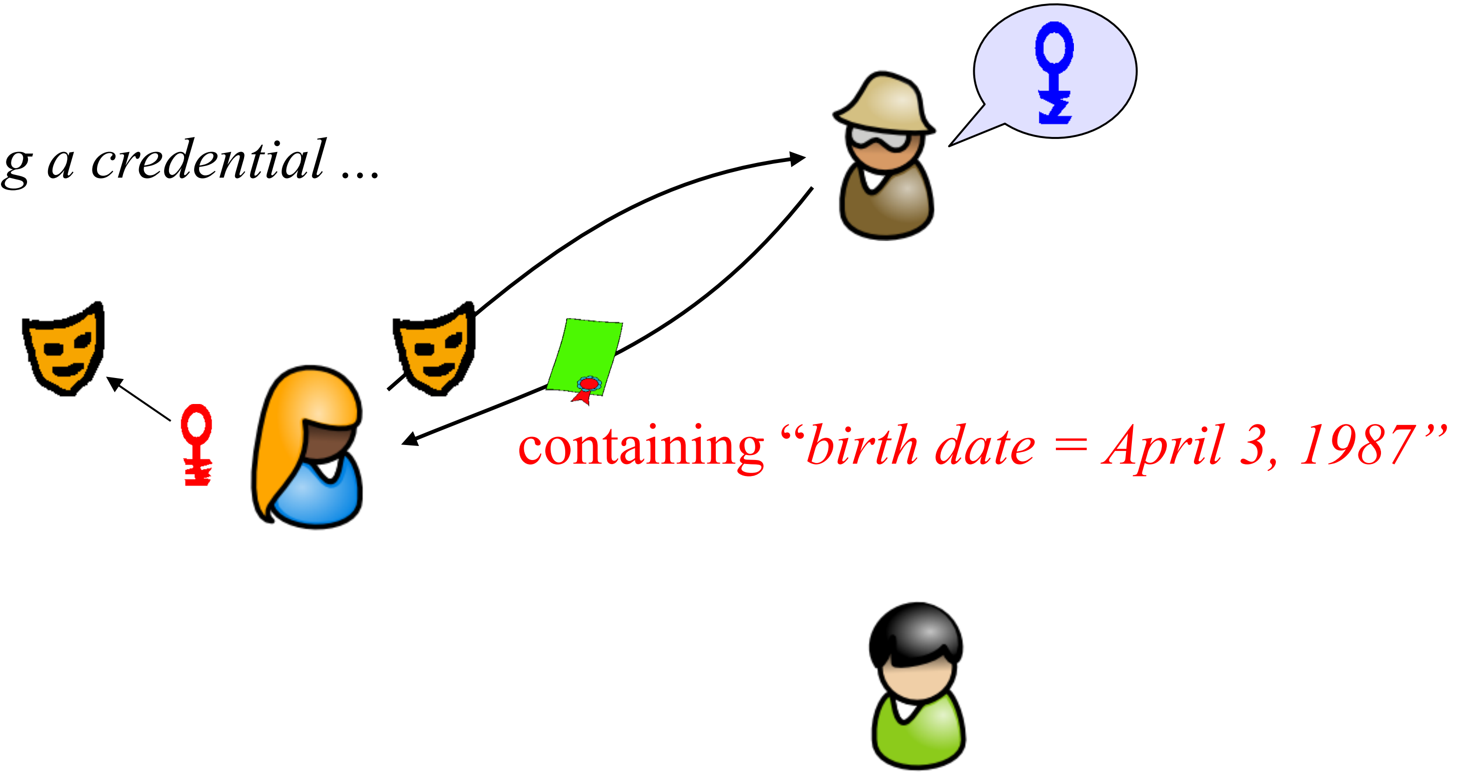


State of the Art: How to Build Them

asking for a credential



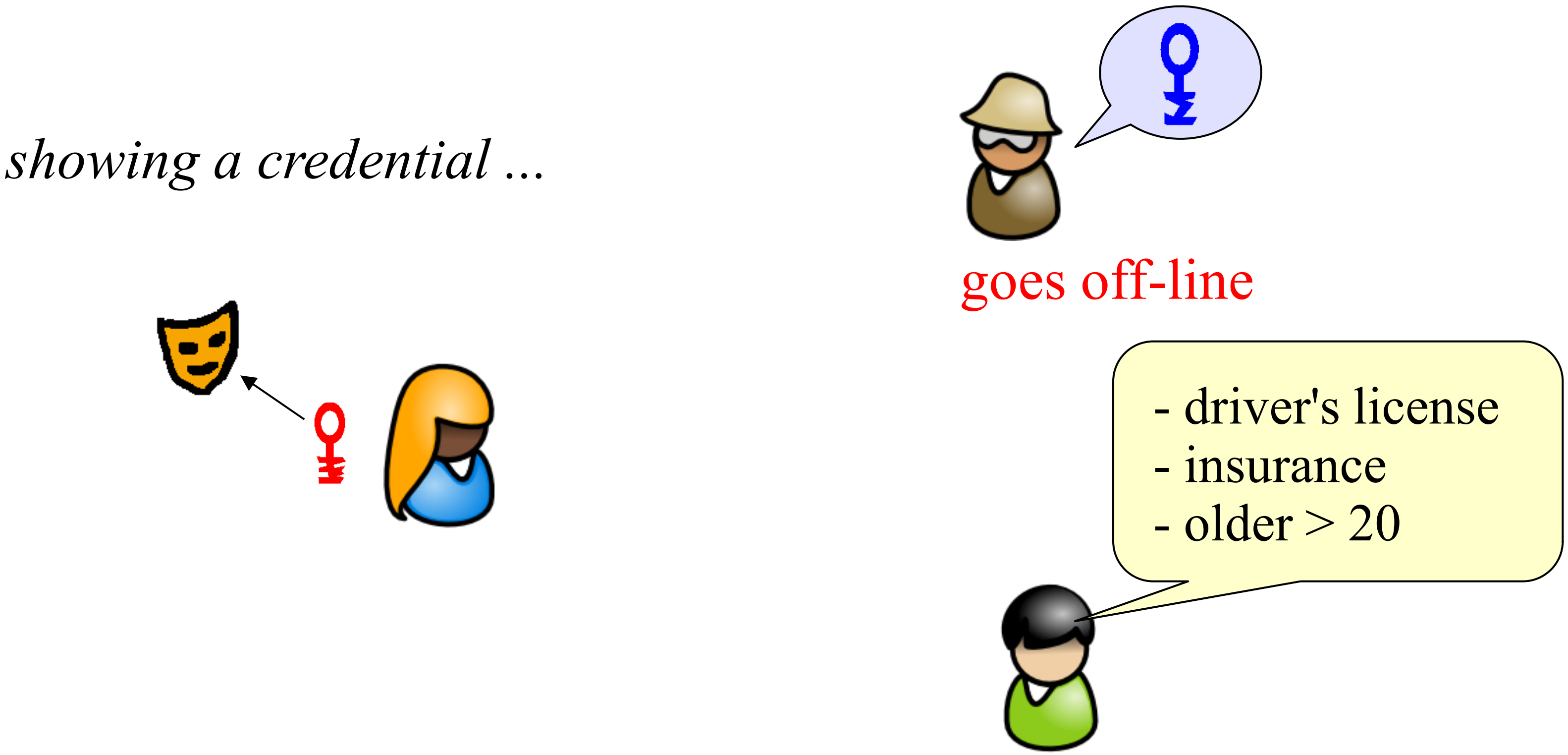
State of the Art: How to Build Them



getting a credential …

containing “*birth date = April 3, 1987*”

State of the Art: How to Build Them



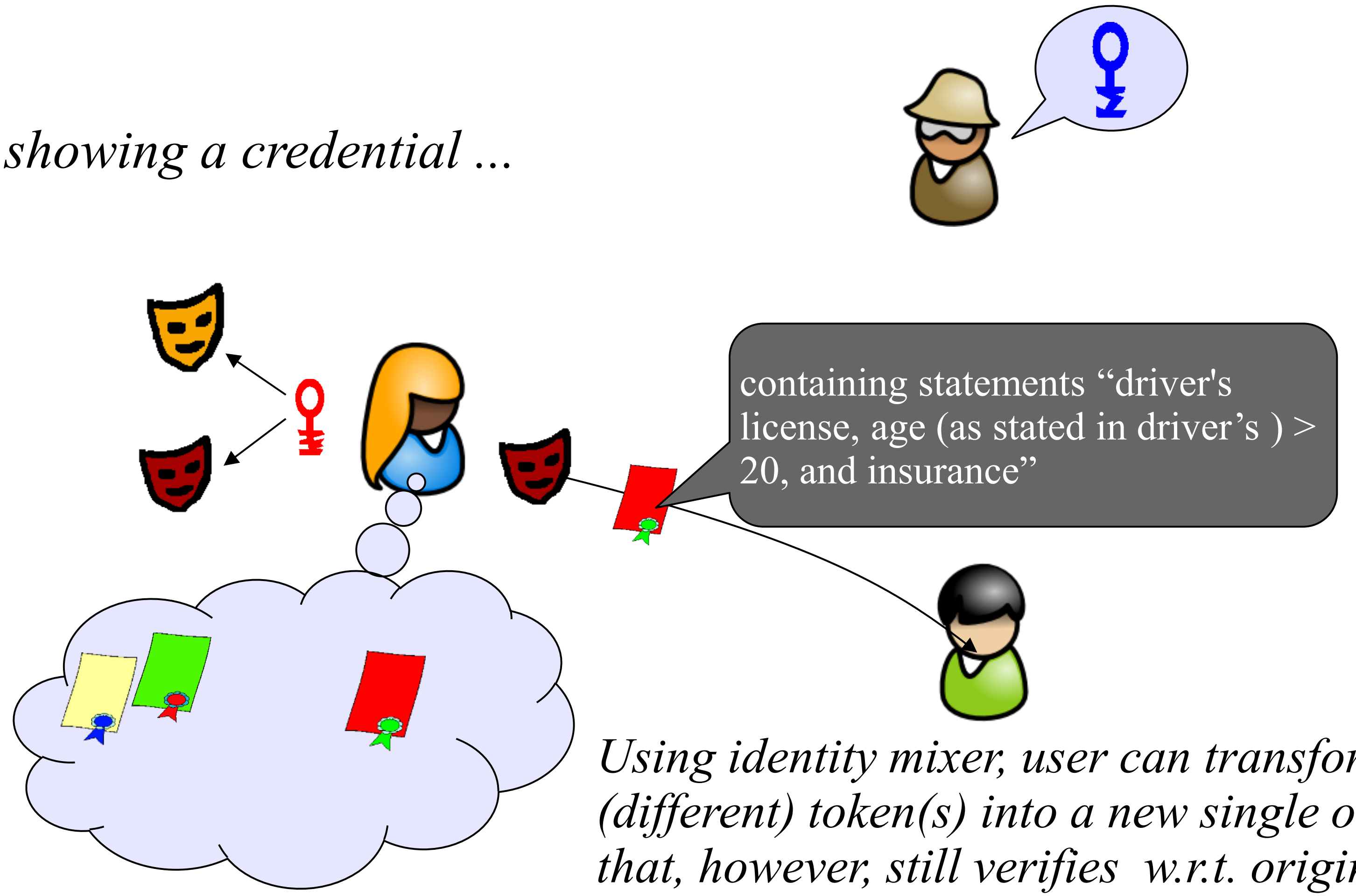
showing a credential …

goes off-line

- driver's license
- insurance
- older > 20

State of the Art: How to Build Them

showing a credential ...

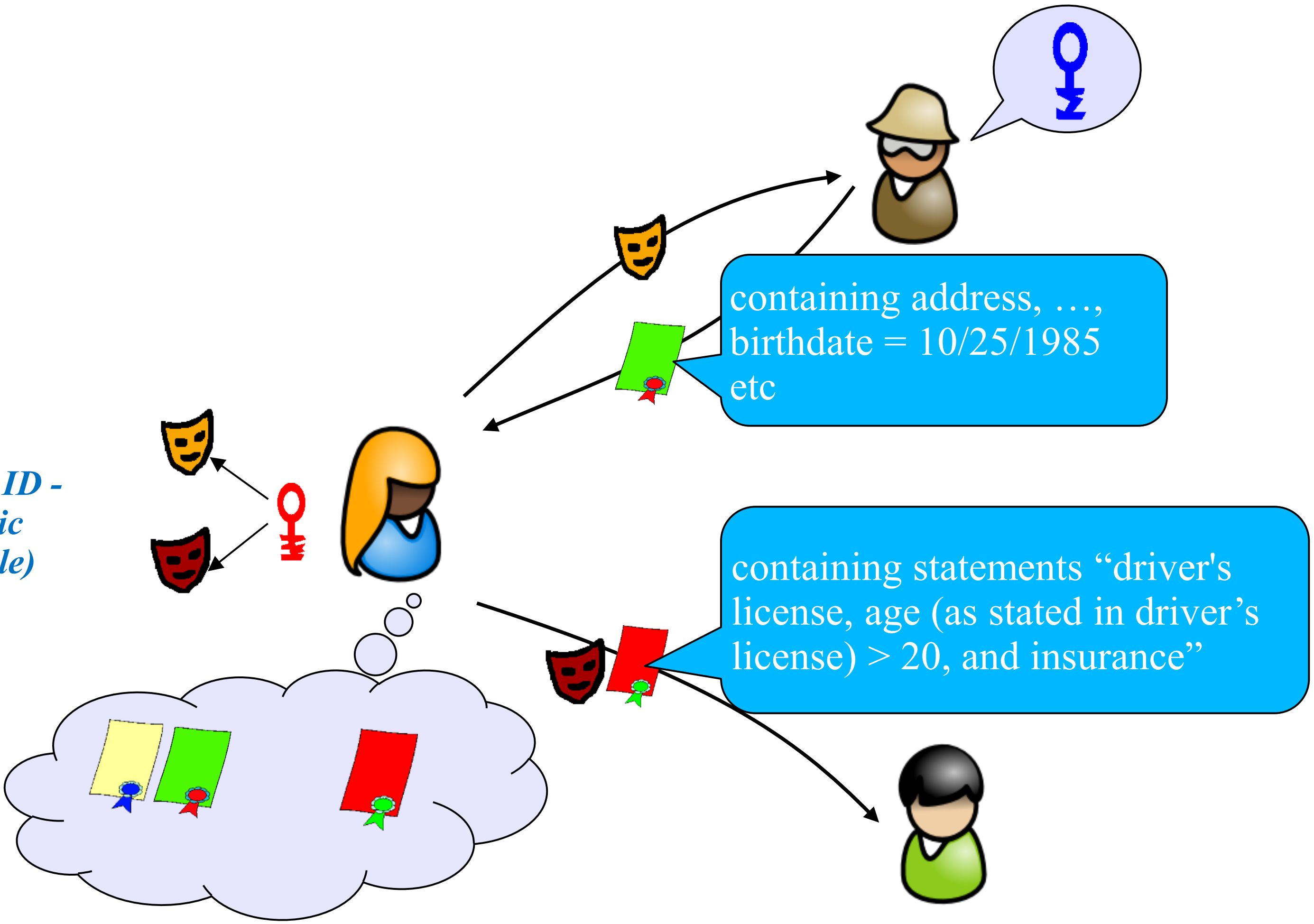


containing statements “driver's license, age (as stated in driver’s) > 20, and insurance”

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.

Anonymous Credentials

One secret ID - Many public (unlinkeable)

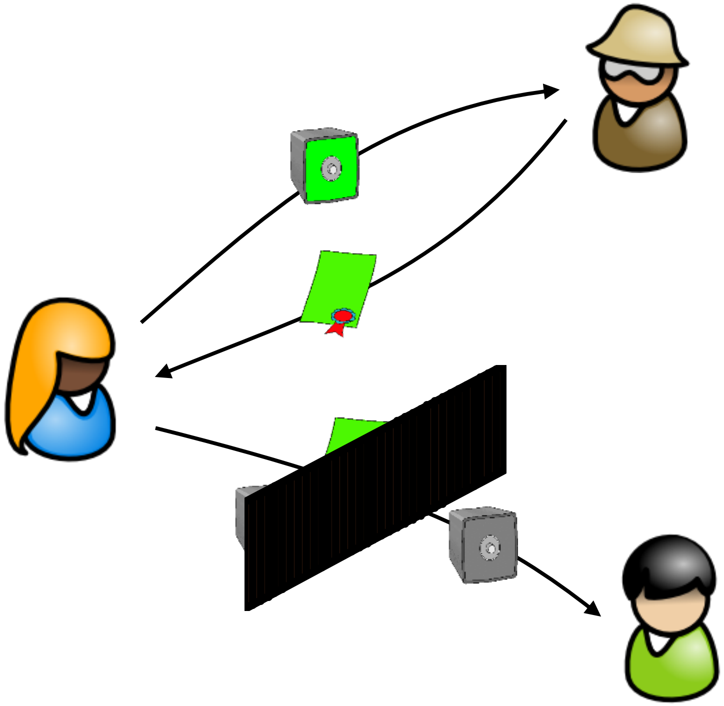


containing address, …, birthdate = 10/25/1985 etc

containing statements “driver's license, age (as stated in driver’s license) > 20, and insurance”

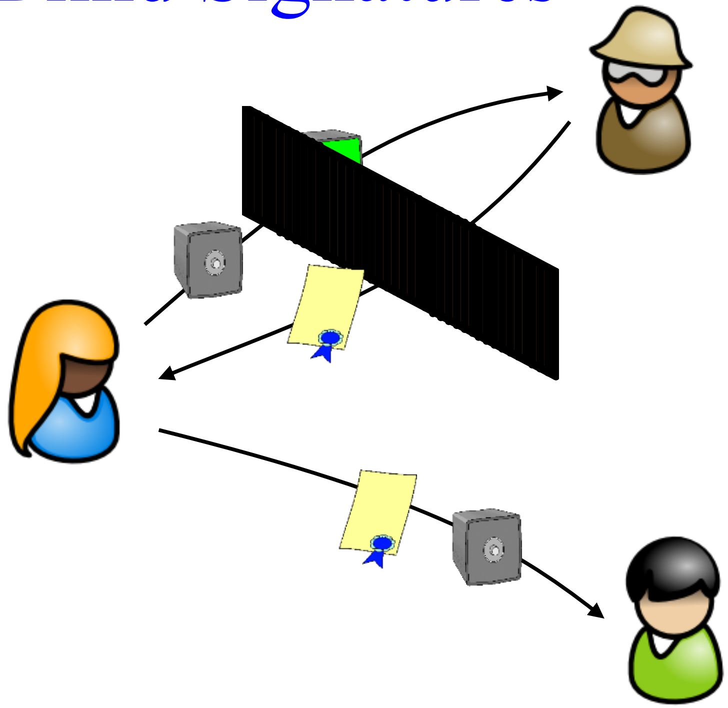
Two Approaches

ZK Proofs



- *can be used multiple times*
- Damgaard,Camenisch&Lysyanskaya
- Strong RSA, DL-ECC,..

Blind Signatures



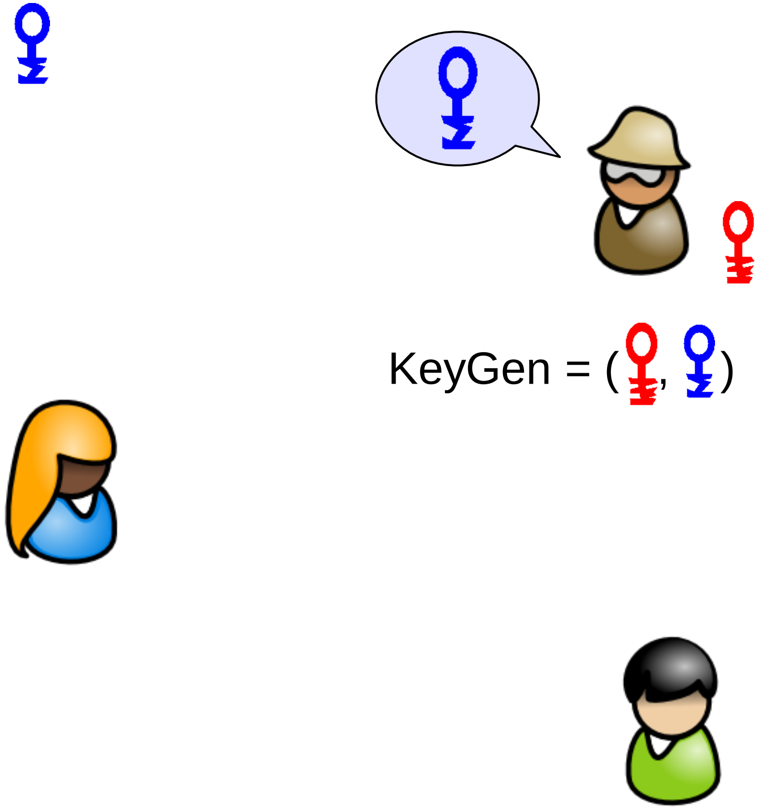
can be used only once

Chaum, Brands, et al.

Discrete Logs, RSA,..

Signature Scheme: Algorithms

Key Generation:
Signer generates
- public key (verification key) 🔑
- secret key (signing key) 🔑



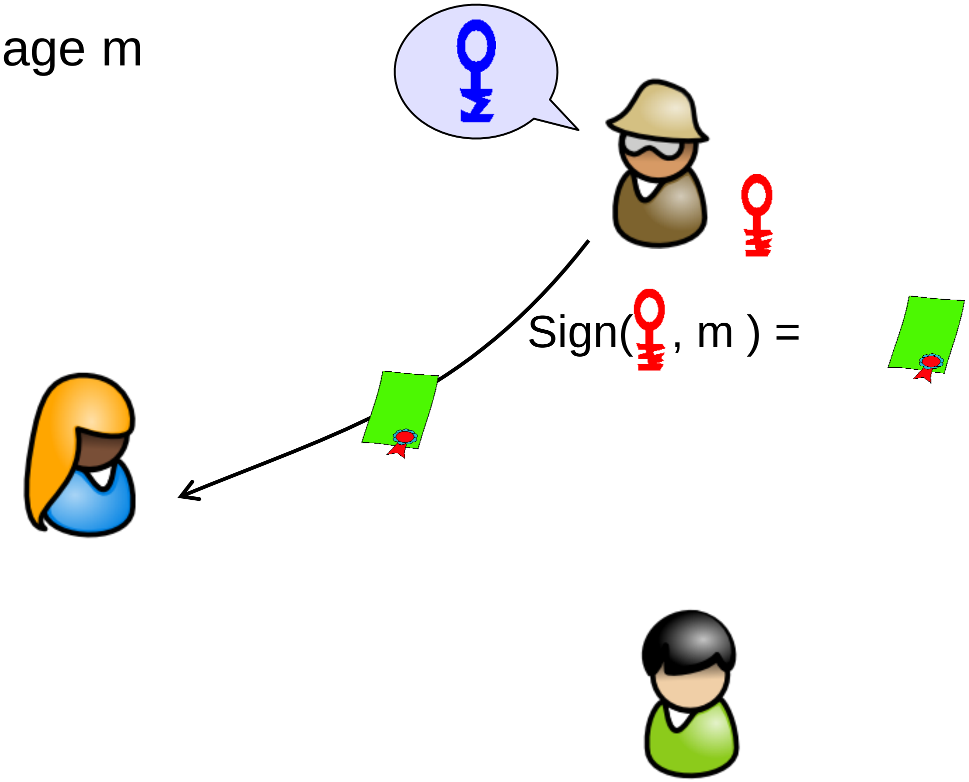
KeyGen = (🔑, 🔑)

Signature Scheme: Algorithms

Signing Algorithm

Signer signs message m

- input: secret key 🔑 and message m
- output: signature 📜 on m



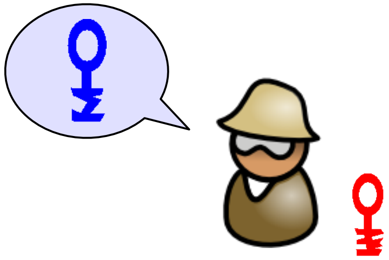
Sign(🔑, m) = 📜

Signature Scheme: Algorithms

Verification Algorithm

Anyone can verify signature on a message

- input: public key 🔑, signature 📜 , and message m
- output: yes/no



Verify(🔑, 📜) = yes

Verify(🔑, 📜) = yes

(remark: no privacy yet ....)

Digital Signature Schemes for Privacy

- Sign blocks of messages m1, … , mk
- Compatible with proof protocols
- Some known schemes:
  - Brands/U-Prove (Discrete Log/Blind Signature)
  - Camenisch-Lyskanskaya (Strong RSA)
  - Camenisch-Lyskanskaya (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.)

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

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

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

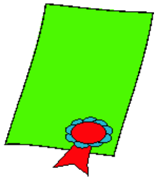
To sign k messages $m1, \ldots, mk \in \{0,1\}^{\ell}$:

□choose random prime $e > 2^{\ell}$ and integer $s \approx n$

□compute c such that

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

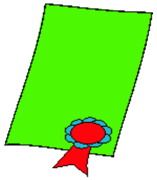
□signature is (c,e,s)



Signature Scheme based on SRSA [CL01]

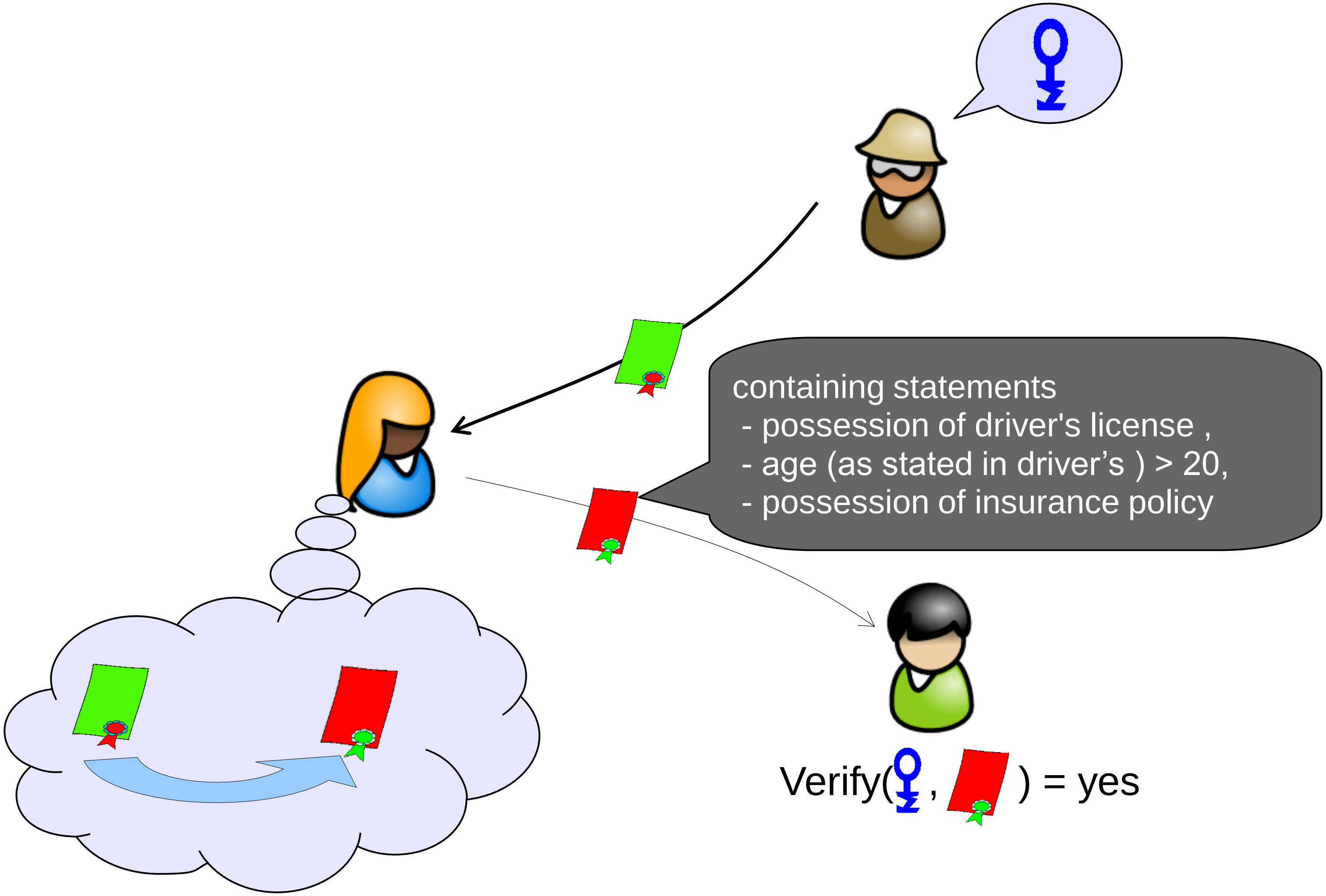
A signature (c,e,s) on messages $m1, \ldots, mk$ is valid iff:

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



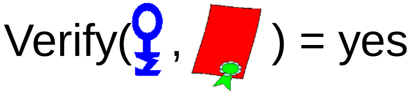
Theorem: *Signature scheme is secure against adaptively chosen message attacks under Strong RSA assumption.*

Recall Goal...



containing statements

- possession of driver's license ,
- age (as stated in driver’s) > 20,
- possession of insurance policy

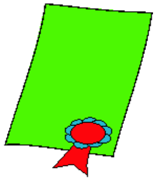


Verify(,) = yes

Recall Verification of Signature

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

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



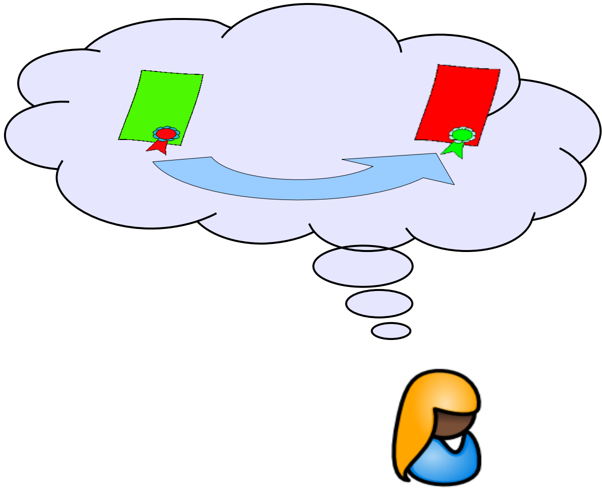
Thus to prove knowledge of values

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

such that the above equations hold.

Problem: c is not an exponent...

Proof of Knowledge of a CL Signature



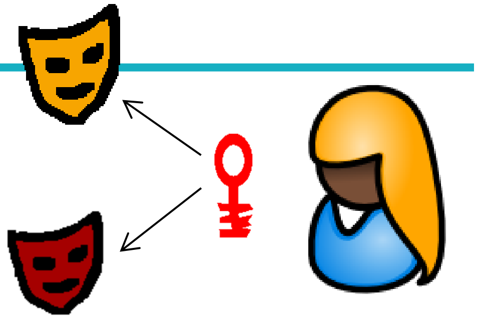
Solution randomize c :

–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'
  - □ $PK\{(e, m1, \ldots, mk, s) : \quad d = c'^{e}\, a_1^{m1} \cdot \ldots \cdot a_k^{mk}\, b^{s} \wedge mi \in \{0,1\}^{\ell} \wedge e \in 2^{\ell+1} \pm \{0,1\}^{\ell} \}$

(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 $P = g^{sk}h^{r}$.
- Domain pseudonym: let $g_d = H(\text{domain})$. Then compute $P = g_d^{sk}$.

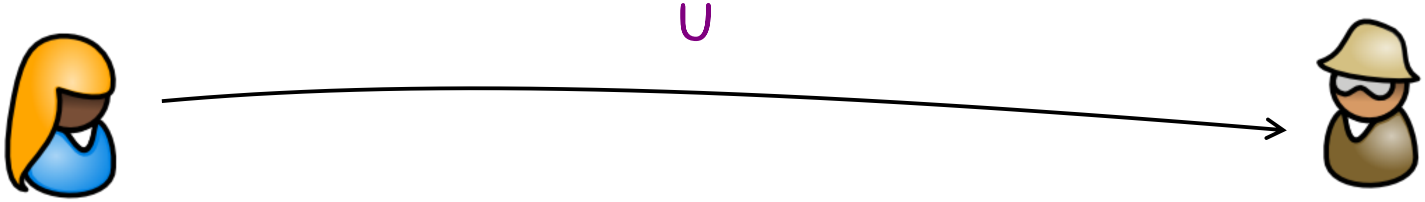
Thus domain pseudonym as unique (per secret identity)

Security:

- Pseudonyms are perfectly unlinkeable.
- Domain pseudonyms are unlinkeable provided
  - Discrete logarithm assumption holds and
  - $H(\text{domain})$ is a random function.

Issuing a Credential to Hidden Messages (idemix)

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

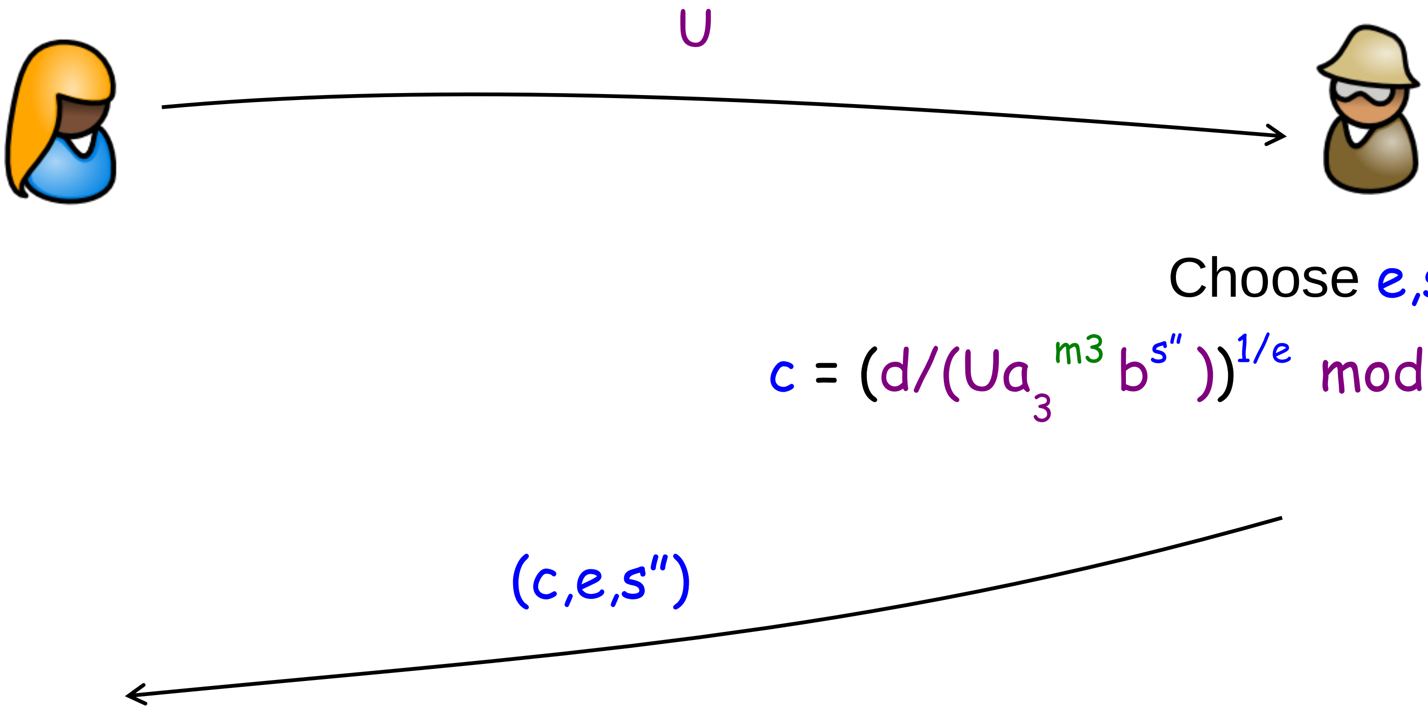


U

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)

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

P

Issuing a Credential to a Pseudonym (idemix)

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

P

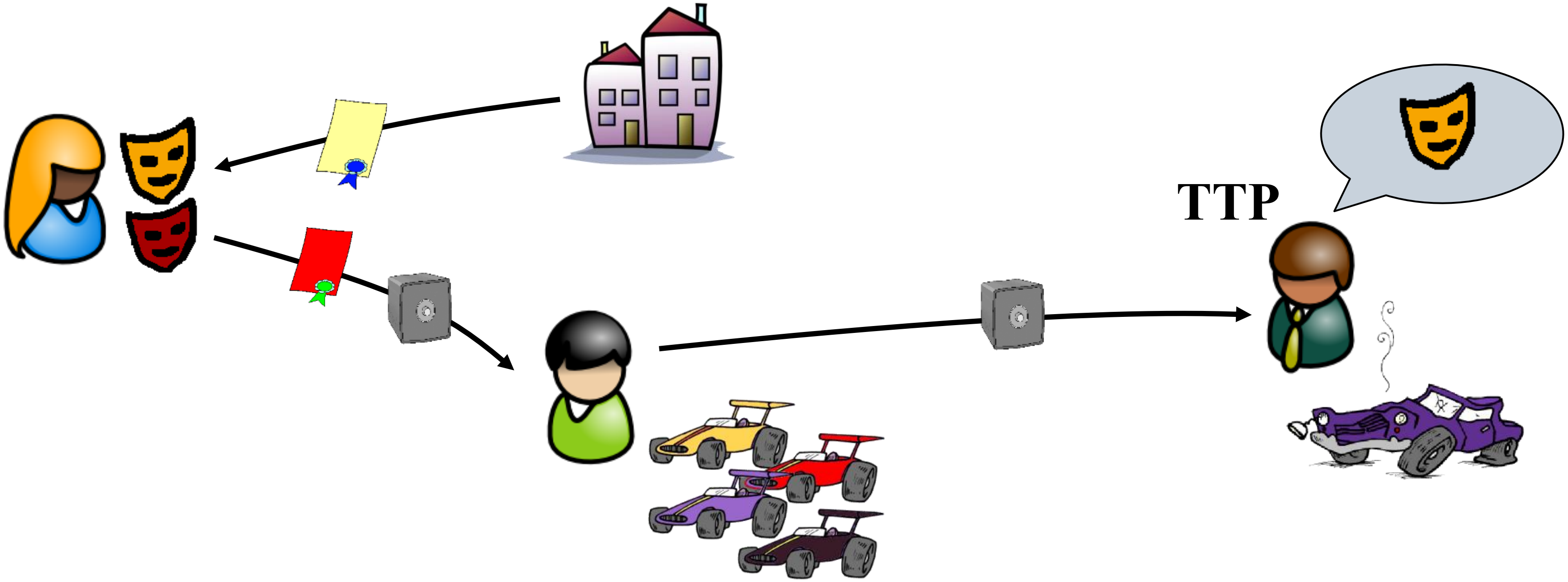
U

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

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

.... and then issue credential just as before

Other Properties: Attribute Escrow (Opt-In)

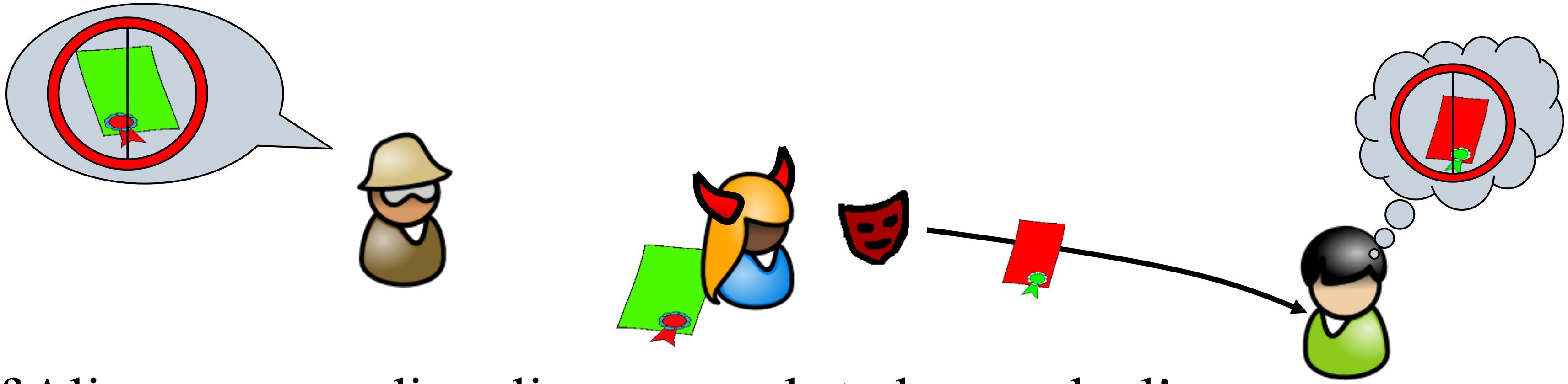


If car is broken: ID with insurance needs be retrieved

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

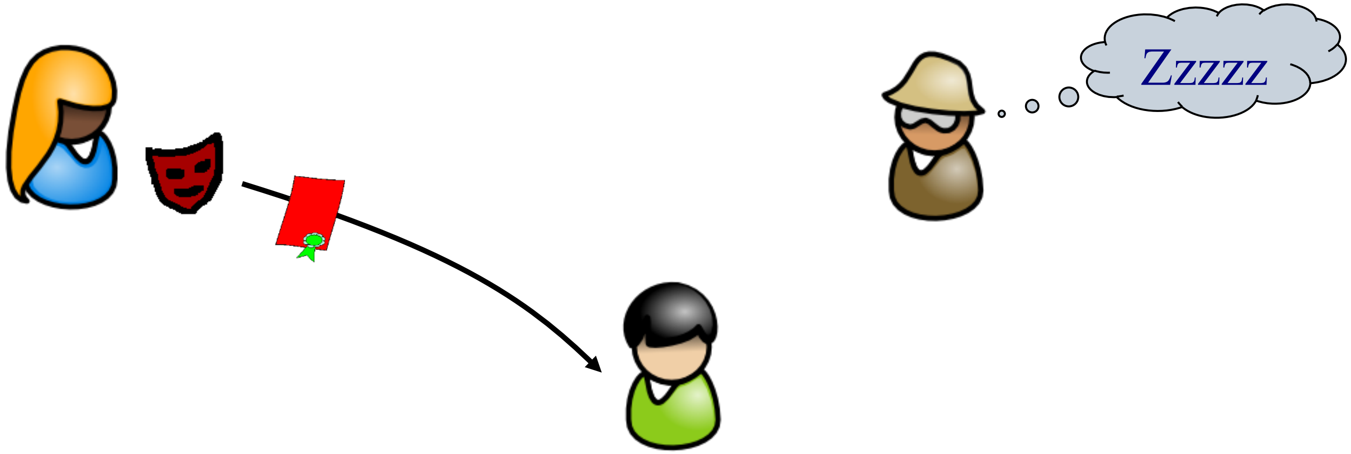
Variants of CRL work (using crypto to maintain anonymity)

- Accumulators
- Signing entries & Proof, ...

Limited validity – certs need to be updated

... For proving age, a revoked driver's license still works

Other Properties: Offline Usage



ID providers (issuers) need sleep, too!

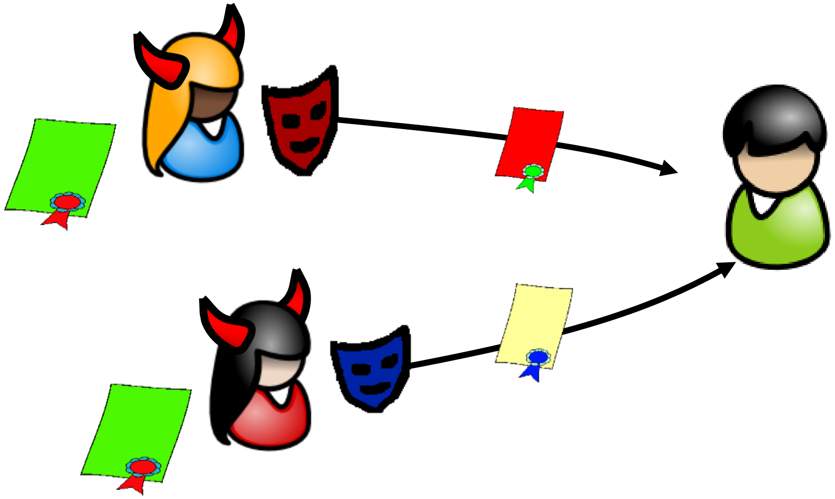
Sometimes it is too expensive to have connectivity

Or a security risk (e.g., ID cards)

Certs can be used as many times as needed!

cf. Revocation; can be done w/ signer's secrets offline

Other Properties: Cheating Prevention



World of Warcraft

Limits of anonymity possible *(optional)*:

If Alice and Eve are on-line together they are caught!

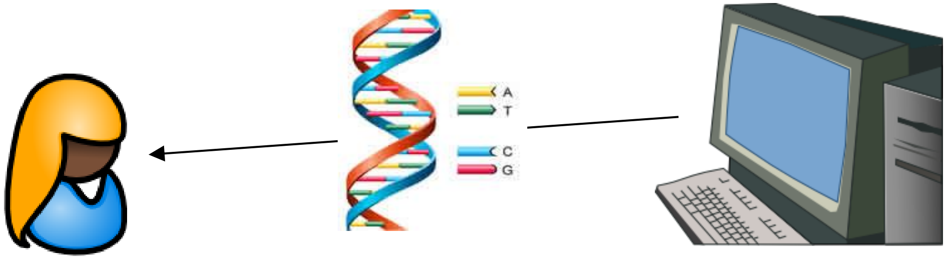
Use Limitation – anonymous until:

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



DNA Database

Simple case: DB learns not who accesses DB

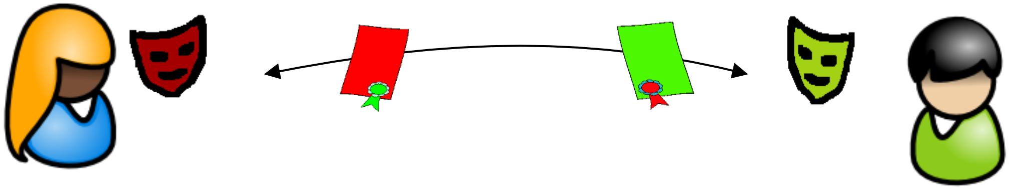
Better: Oblivious Access to Database (OT with AC)

Server must not learn *who* accesses

which record

Still, Alice can access only records she is *authorized* for

Secret Handshakes



Alice and Bob both define some predicate PA and PB

Alice learns whether Bob satisfies PA if she satisfies PB

Key markets

- E-Government (citizen identities)
- E-Health (health record management)
- Cloud computing (“don’t trust us” cloud providers)
- Document signing (with minimal disclosure)
- Advertising (privacy-respecting ad platform)
- E-Cash
- E-Voting
- Social Networking
- Document signing

Thank you!