

E-Privacy – Privacy in the Electronic Society

Anonymous Credentials I

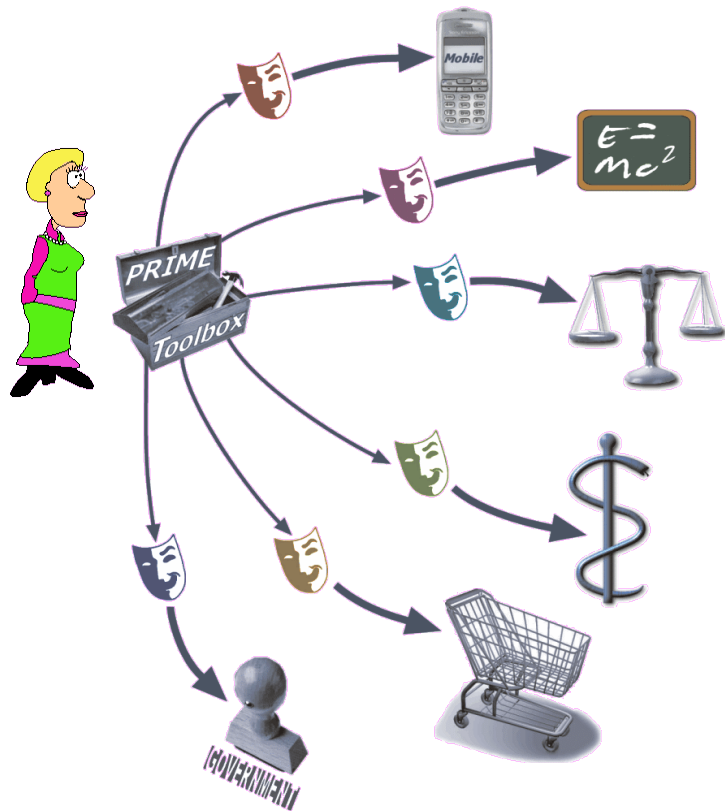
Jan Camenisch

IBM Zurich Research Laboratory

Rüschlikon

jca@zurich.ibm.com

Identity, Privacy, and Trust Management

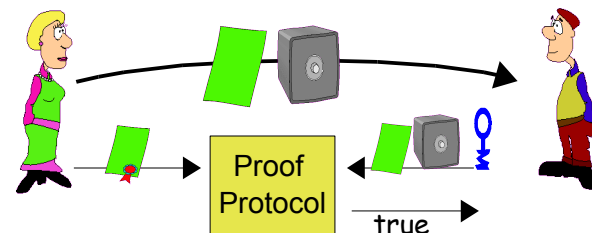
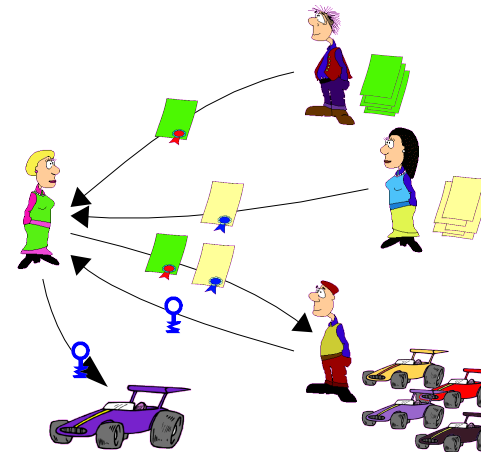


Vision: *In the Information Society, users can act and interact in a safe and secure way while retaining control of their private sphere.*

Goal: Allow the user to manage and protect her private data effectively.

Outline Anonymous Credentials

- Credential Systems
 - Requirements
 - Definition
 - How to prove Security
- Building Blocks
 - Proof Protocols
 - Commitment Schemes
 - Signature Schemes
 - Encryption Scheme
- Putting Them Together

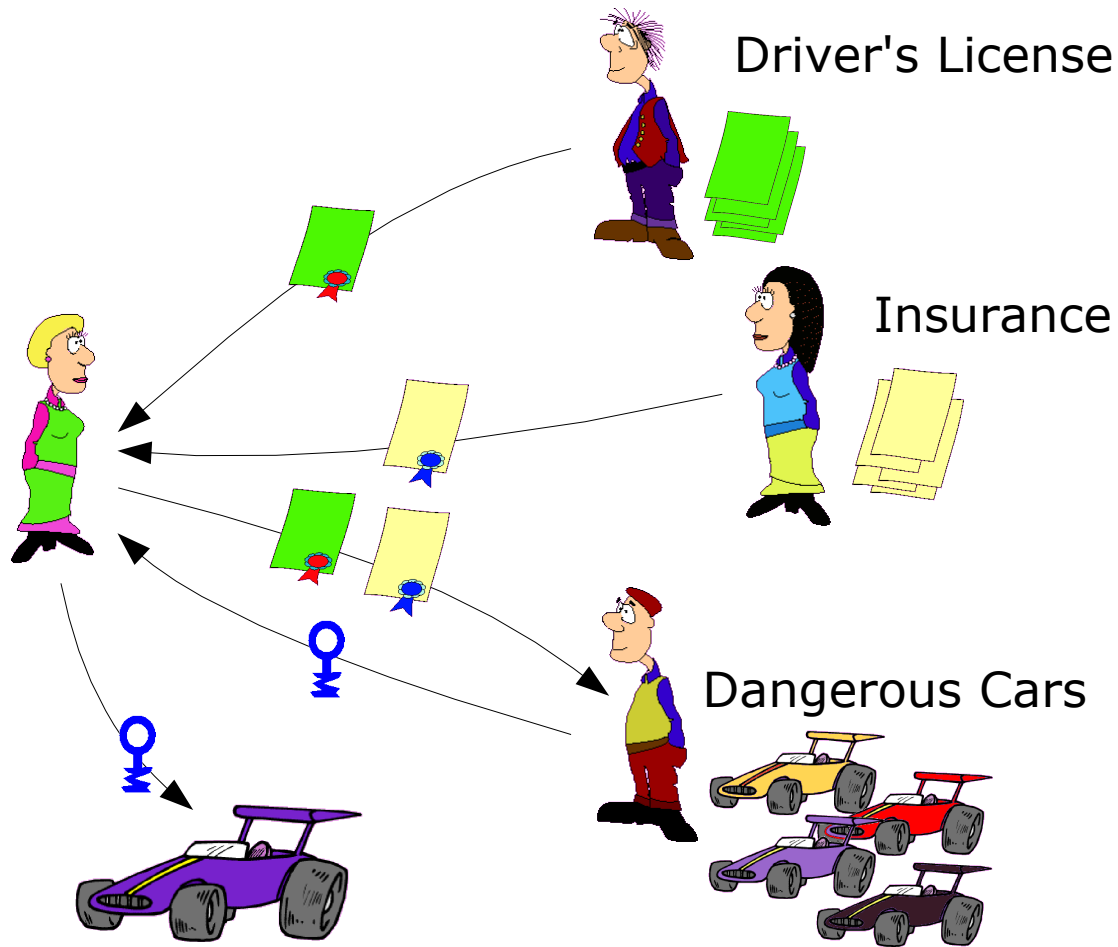


$$c^e = a_1^{m1} \dots a_k^{mk} b^s d \text{ mod } n$$

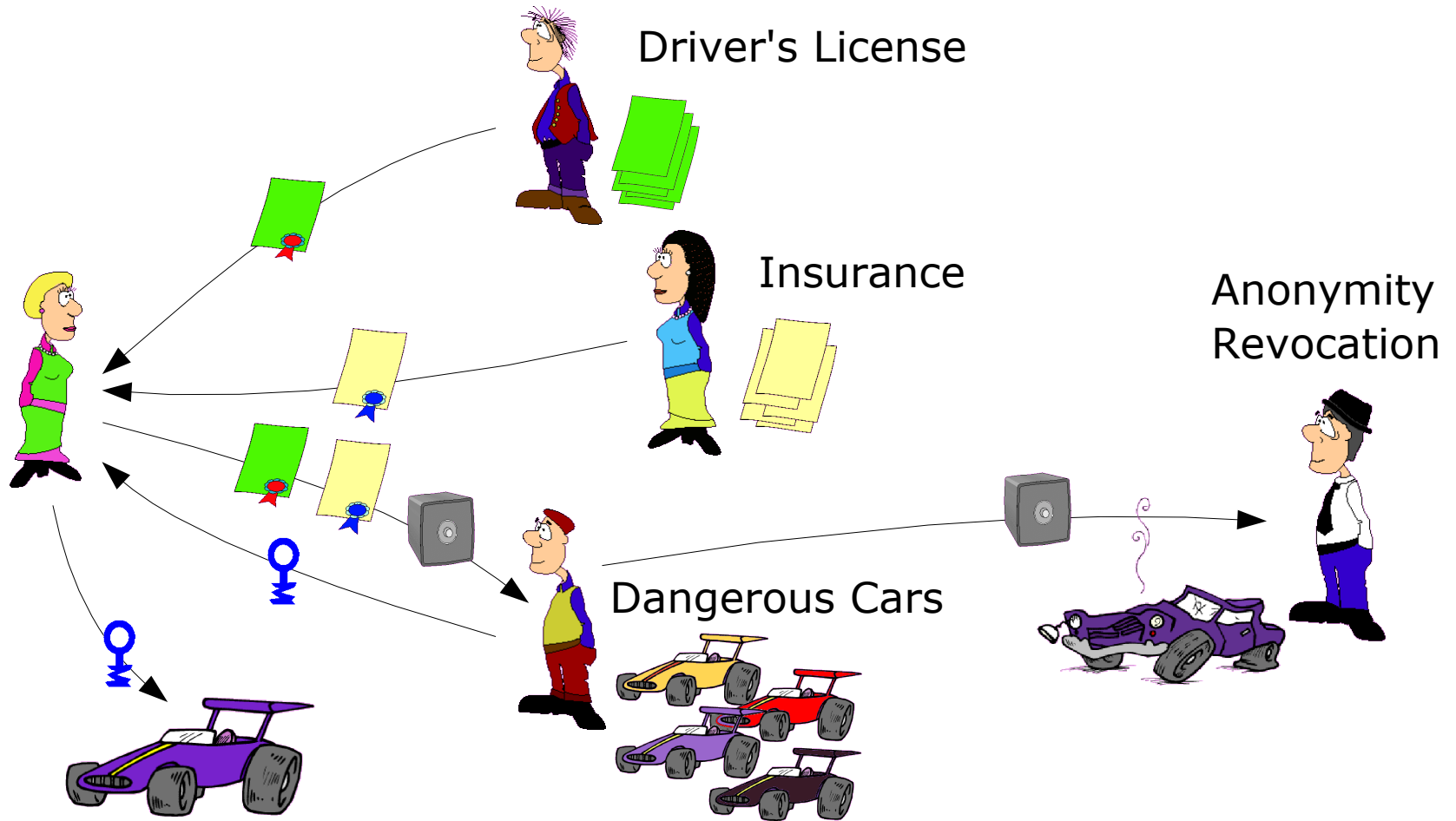
Credential Systems

Requirements

Example

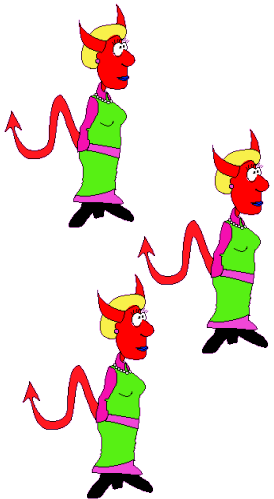


Example cont'



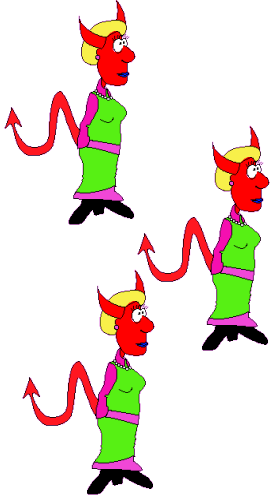
Basic Requirements of Pseudonym System

- Protection of user's privacy
 - anonymity
 - unlinkeability (multi-use)



- Unforgeability of credentials
- Consistency of credentials (no pooling)

Extra Requirements of Pseudonym System



- Sharing of credentials
 - Prevention & Detection

- Anonymity revocation
 - local
 - global

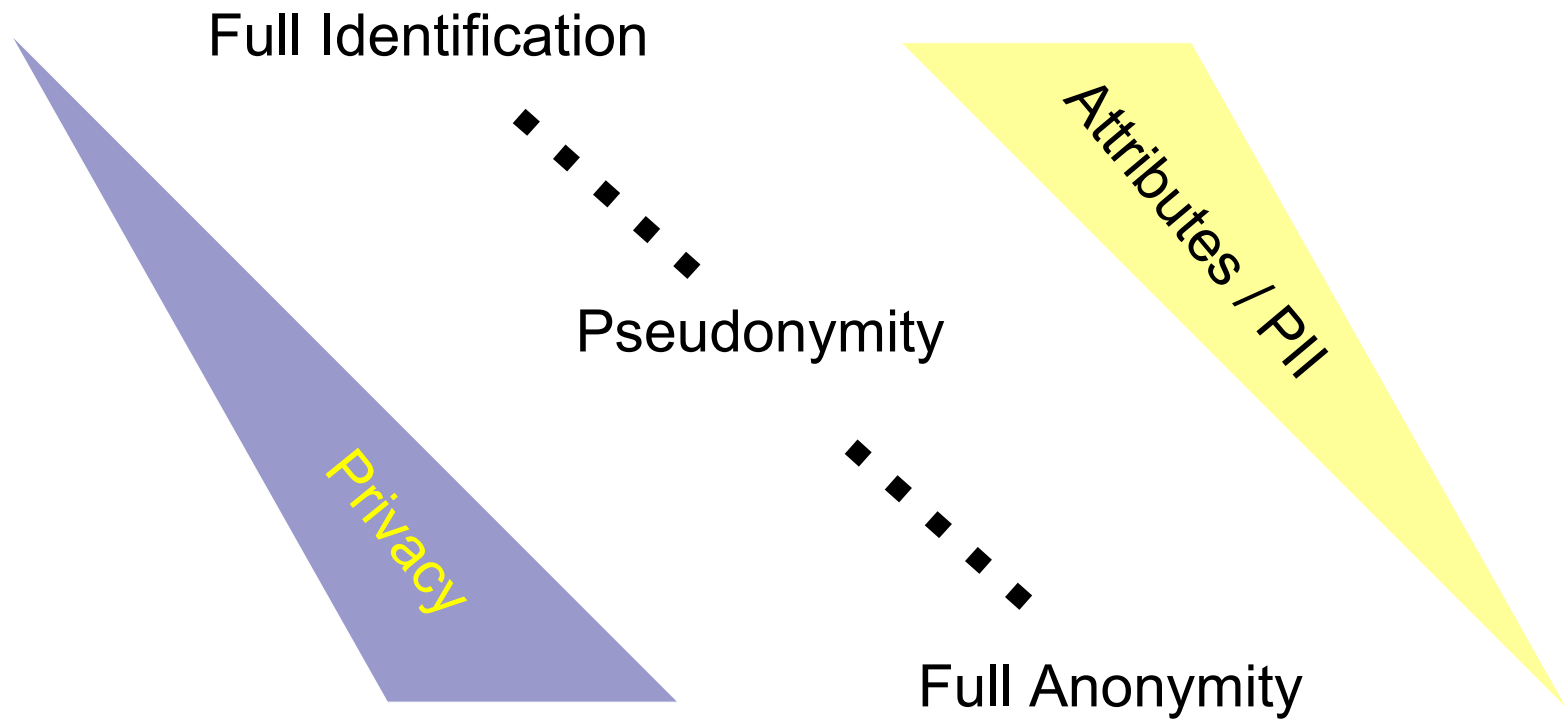
- Revocation of credentials

- Encoding of attributes

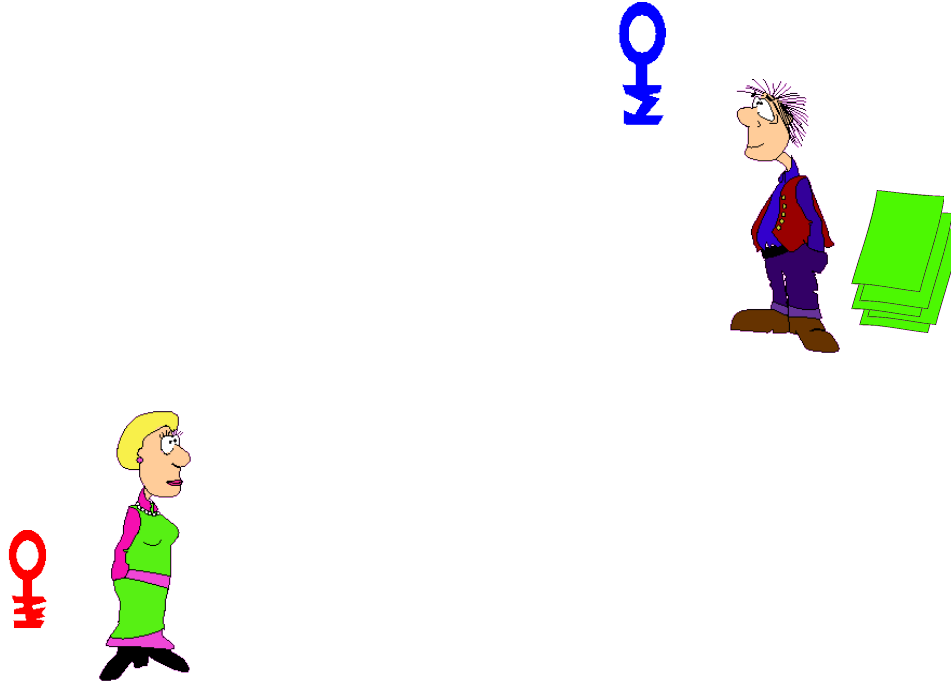
- One-show credential (e-cash)



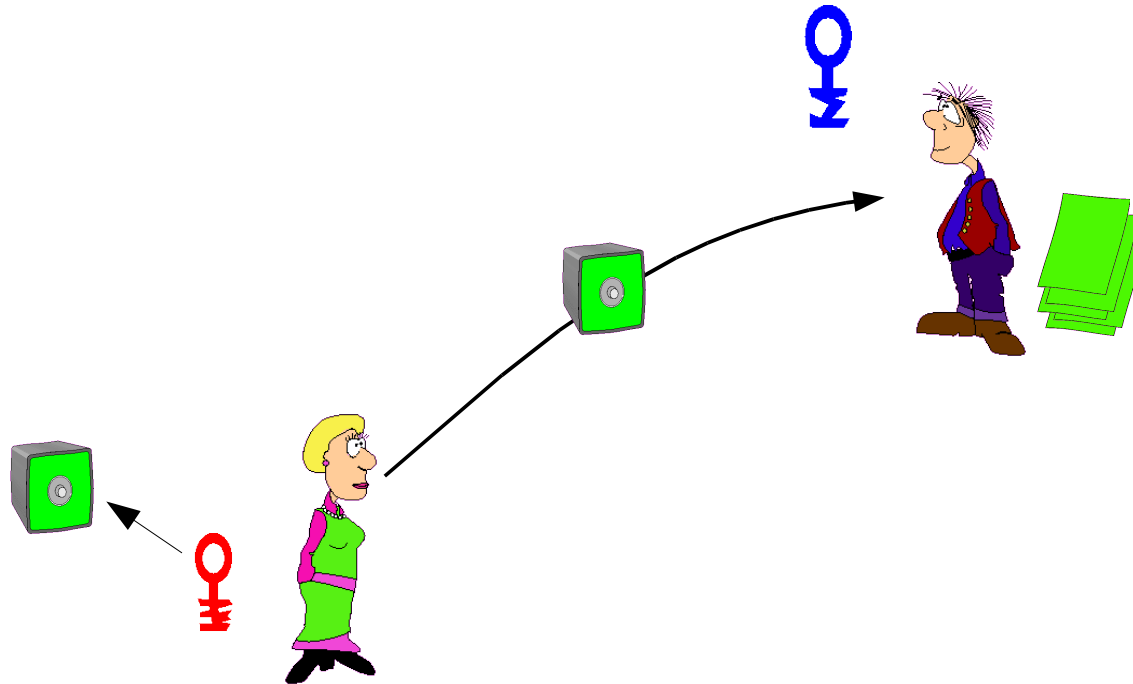
Identification vs. Anonymity



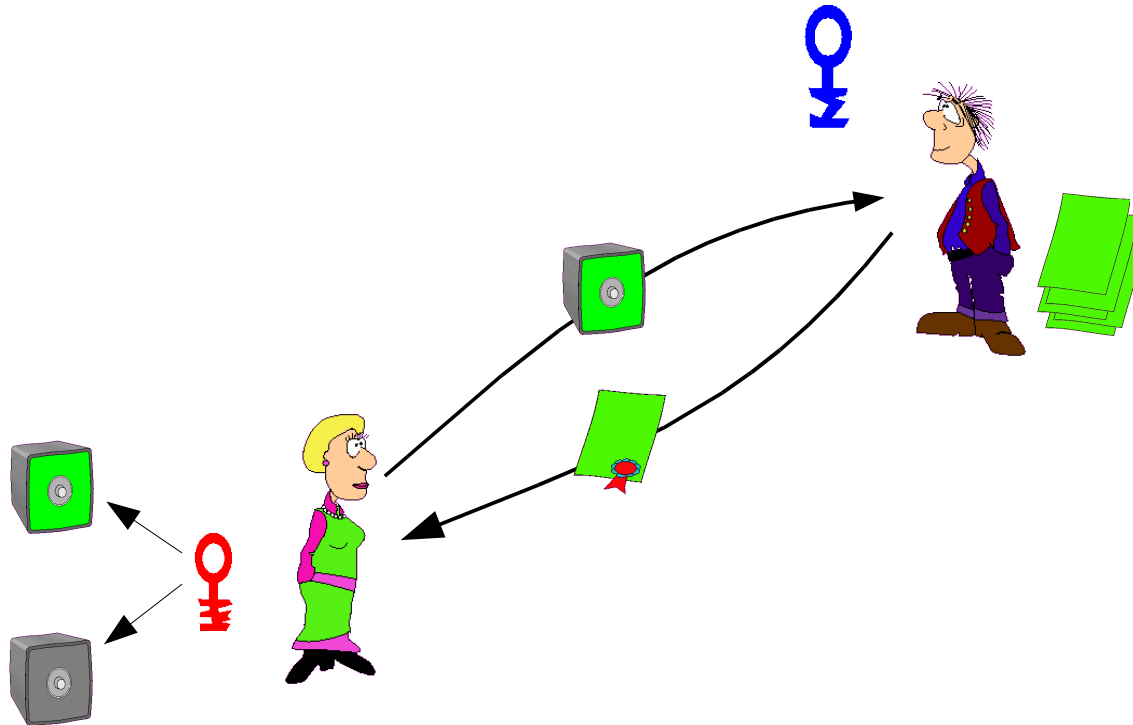
Privacy Enhancing Solution



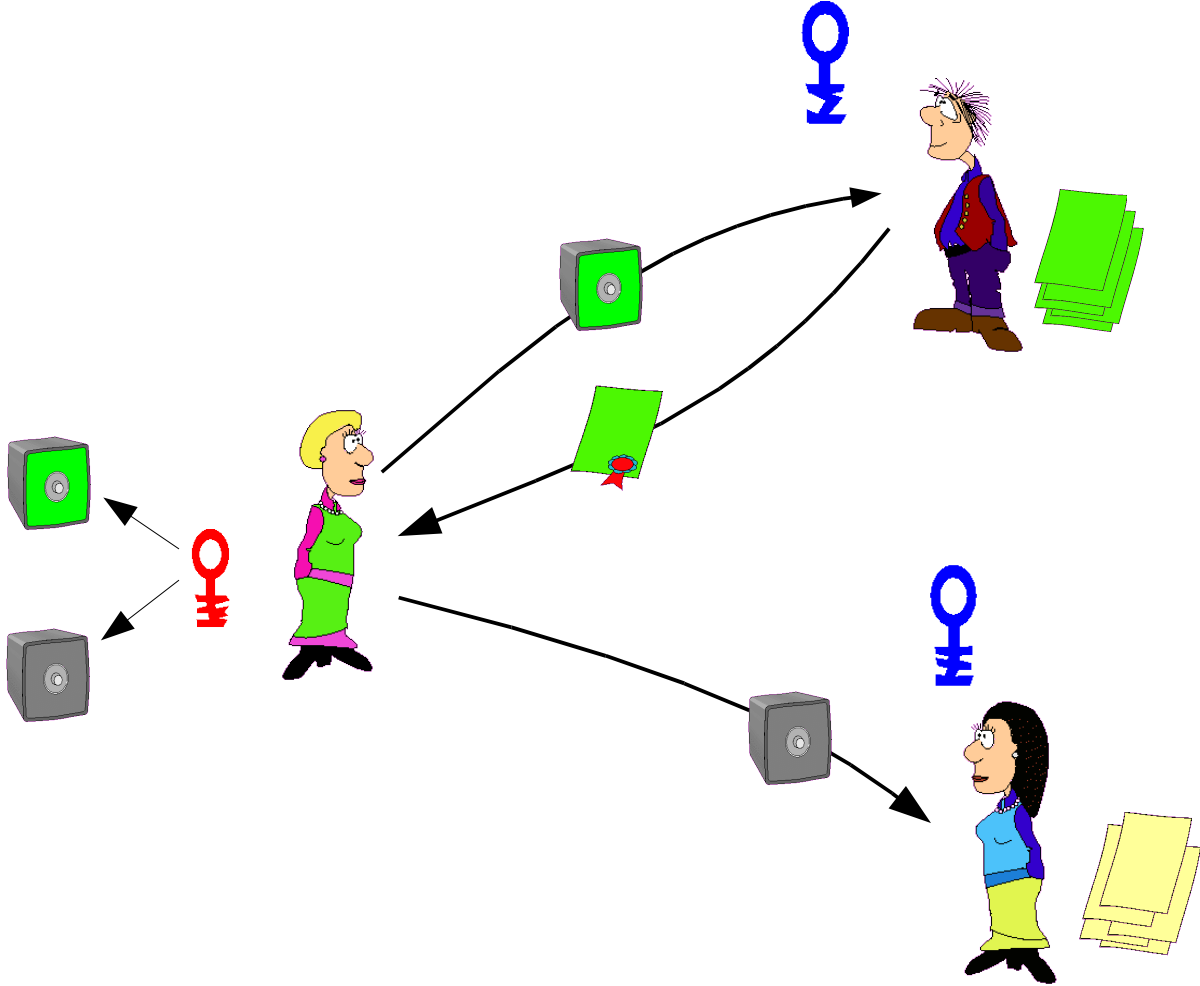
Privacy Enhancing Solution



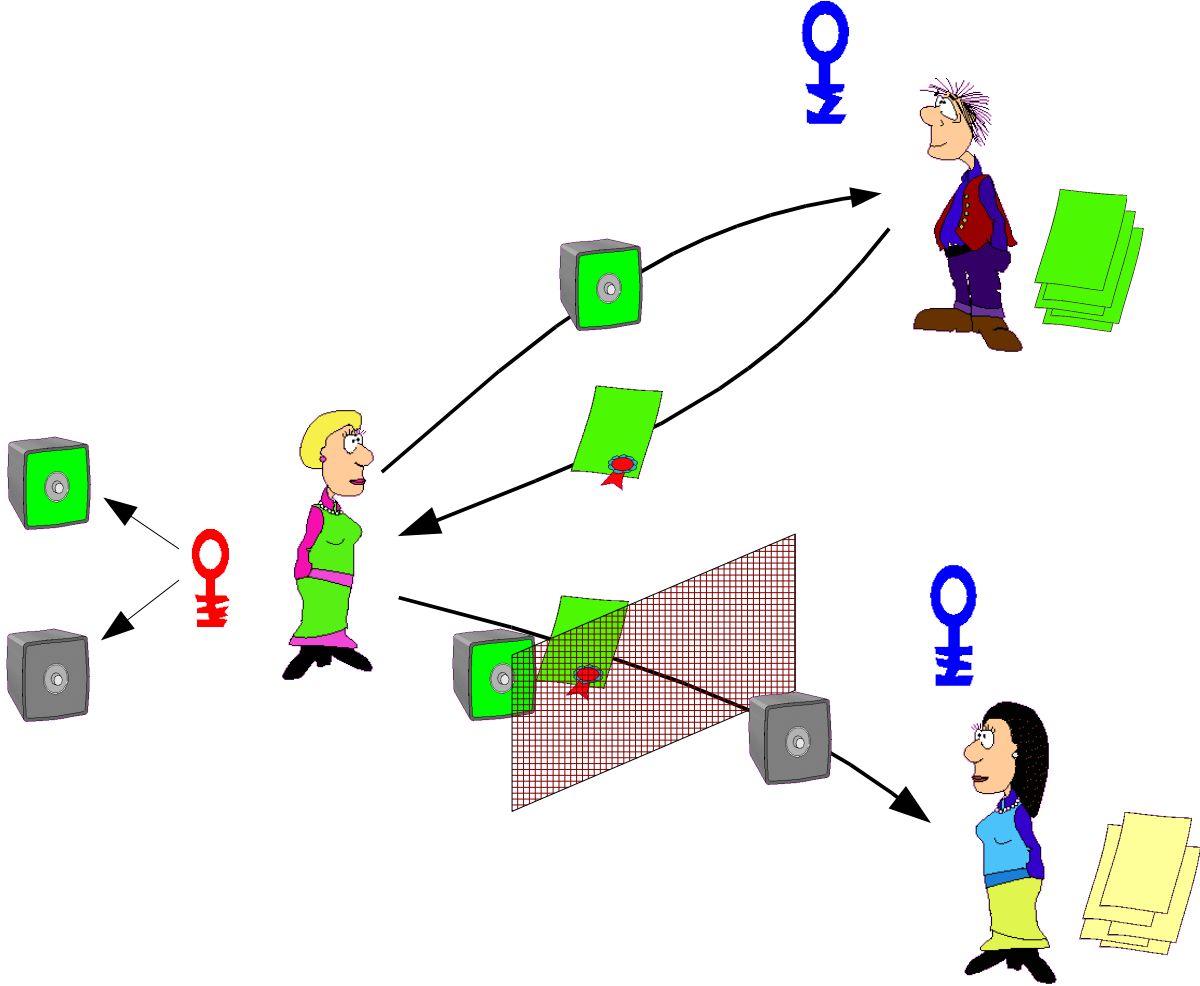
Privacy Enhancing Solution



Privacy Enhancing Solution



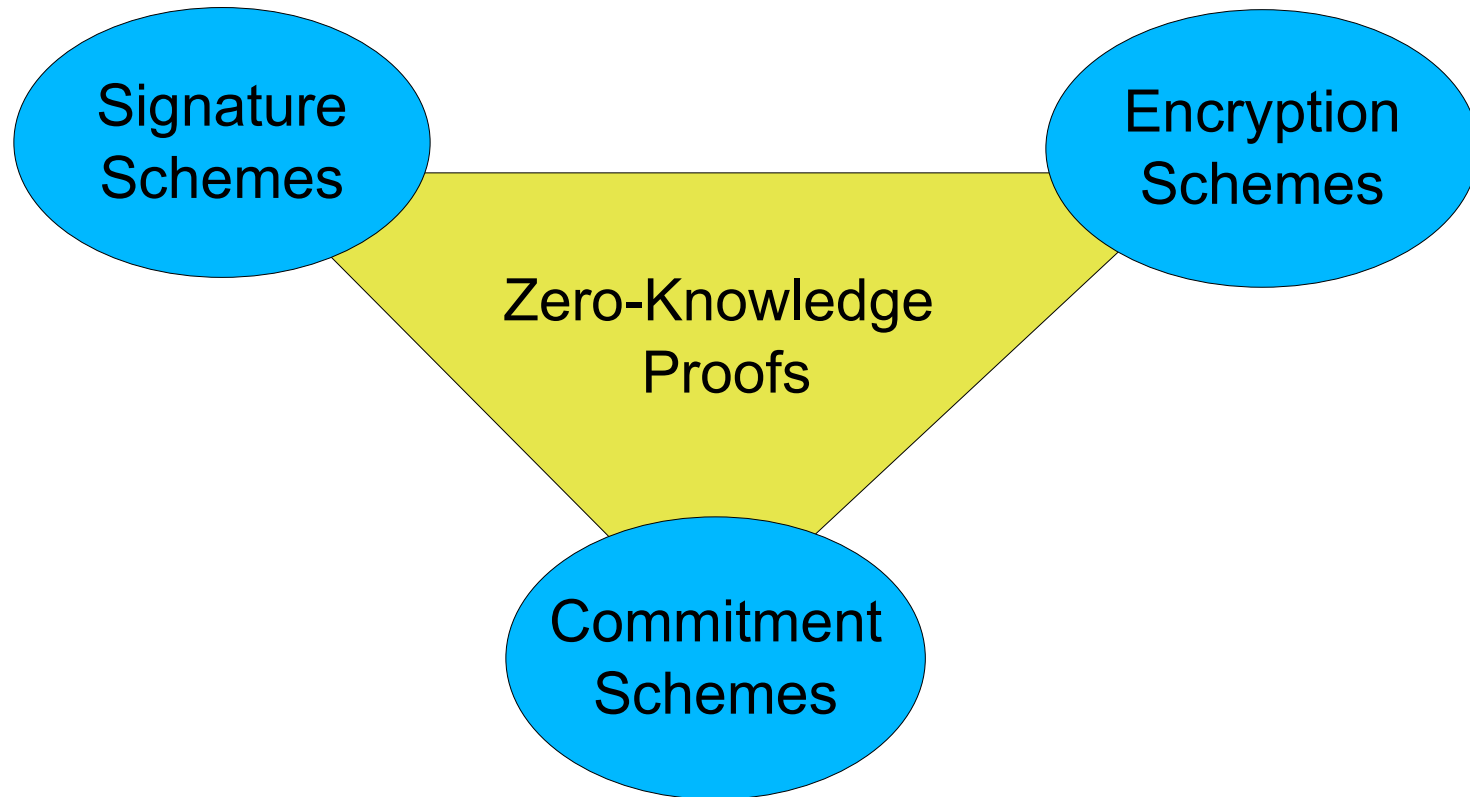
Privacy Enhancing Solution



Privacy Enhancing Solution

- Each Org has public key of signature scheme
- Each user has a **single secret** key but many public keys
- Certificate = signature on user's **secret** key + attributes
 - Attributes might be hidden from issuer
- When using certificate:
 - **Prove knowledge** of certificate instead of revealing it
 - Prove that different certificates contain same secret key
 - Selectively reveal attributes
 - Verifiably encrypt attributes

Required Technologies



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

Zero-Knowledge Proofs of Knowledge

..... for 3-Colorability of Graphs

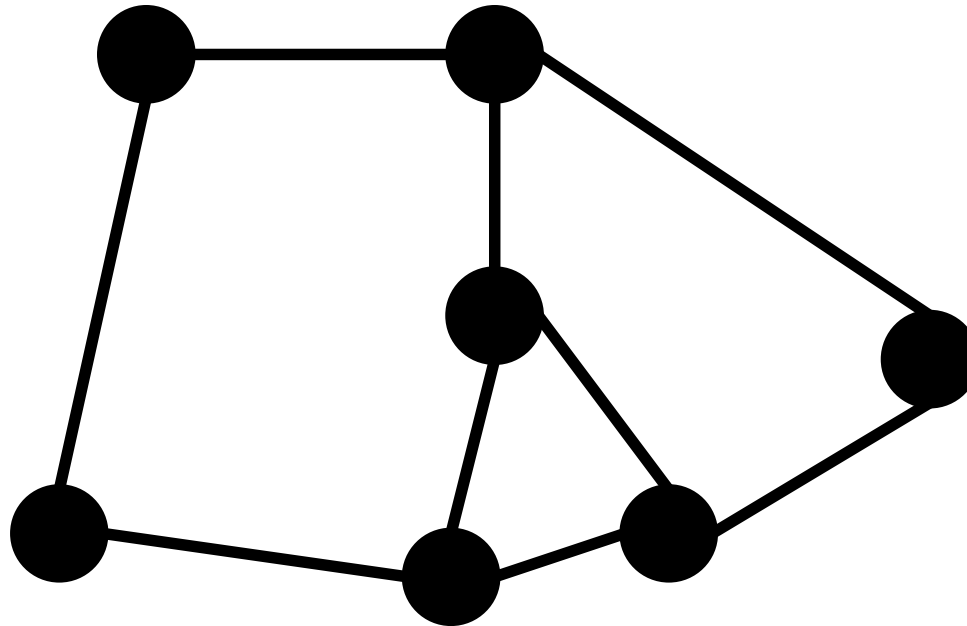
3-Colorability



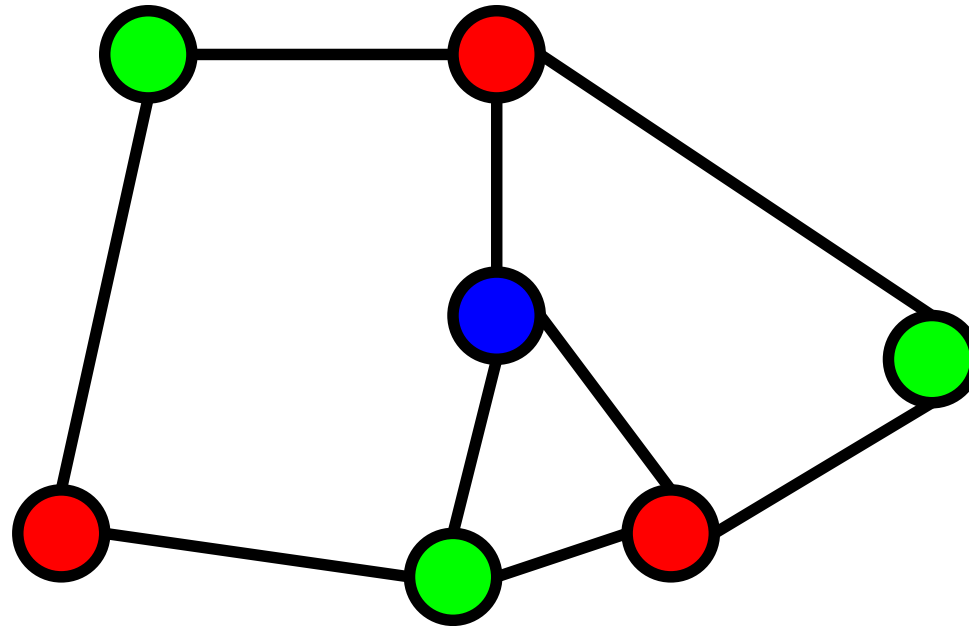
3-Colorability



3-Colorability



3-Colorability



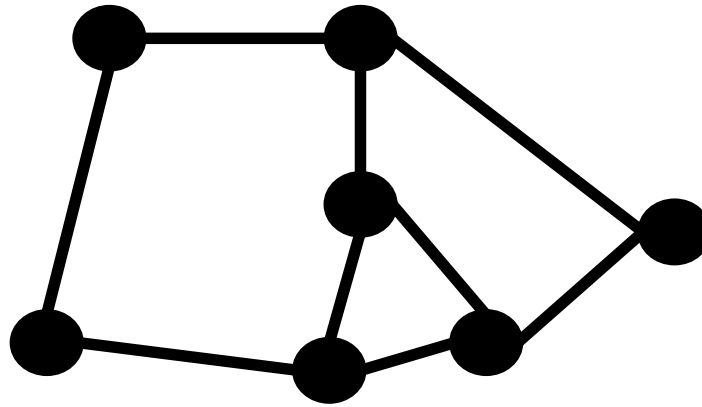
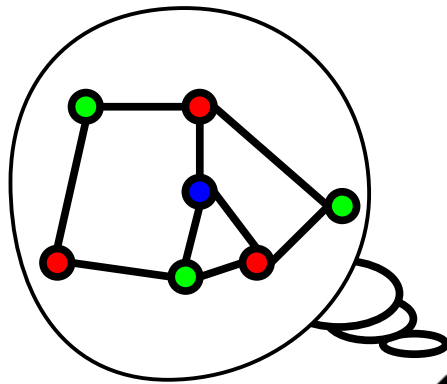
Deciding whether a graph is 3-colorable is hard

Zero-Knowledge Proof

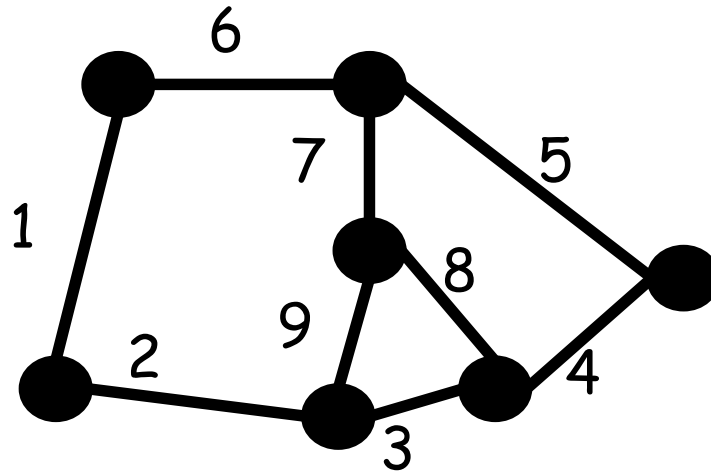
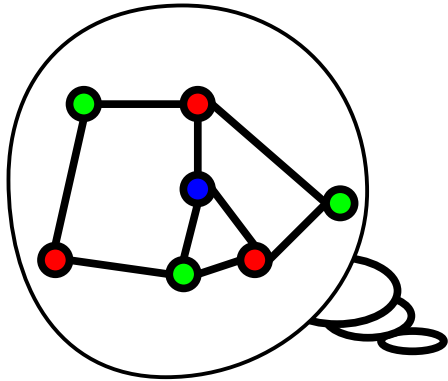
- Prover Alice wants to convince verifier Bob that some theorem is true.
- How can she convince Bob without him Bob learning the proof?



Scenario: Alice & Bob

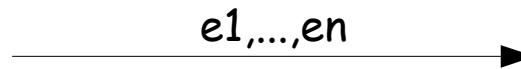


Protocol

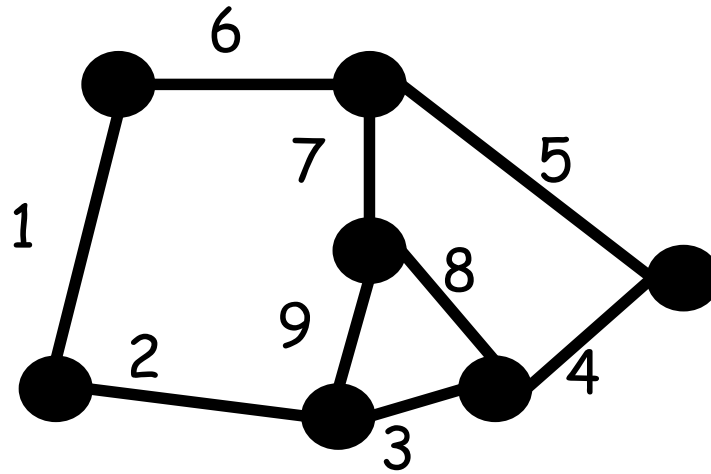
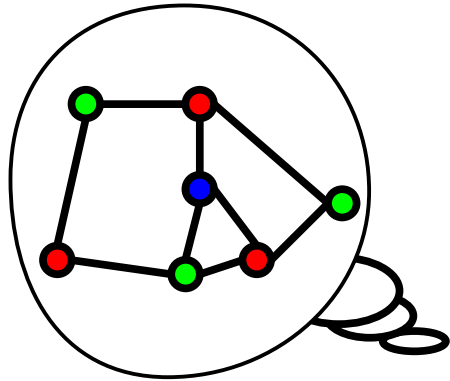


$n = 7$ nodes
 $m = 9$ edges

random $\pi \{r,g,b\} \rightarrow \{r,g,b\}$
for $I = 1, \dots, n$
random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$

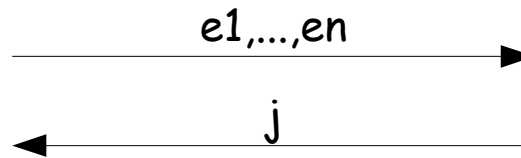


Protocol



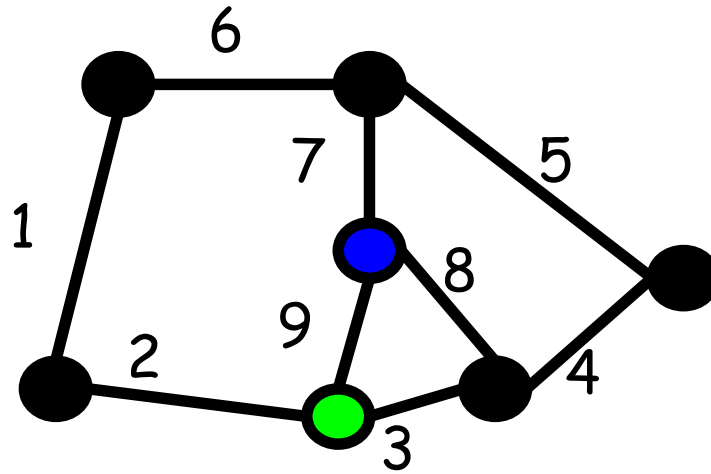
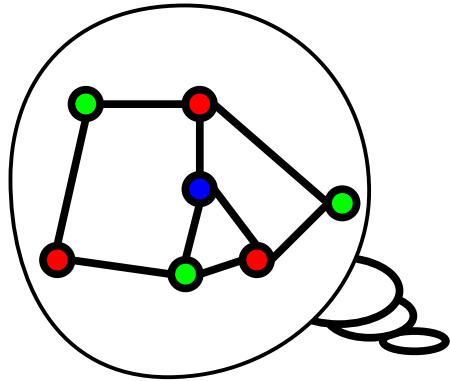
$n = 7$ nodes
 $m = 9$ edges

random $\pi \{r,g,b\} \rightarrow \{r,g,b\}$
for $I = 1, \dots, n$
random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$



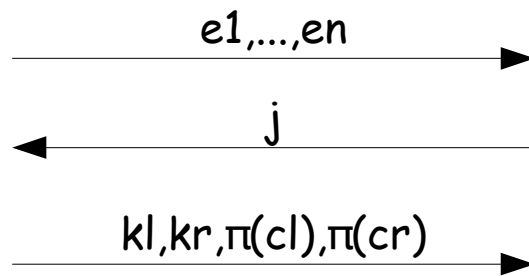
random j

Protocol



$n = 7$ nodes
 $m = 9$ edges

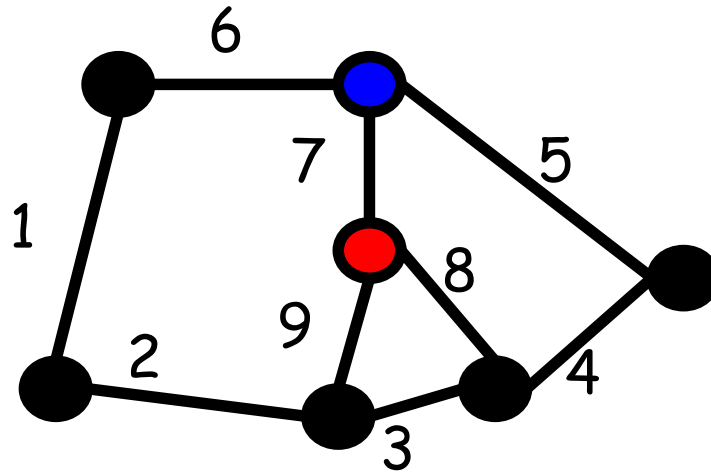
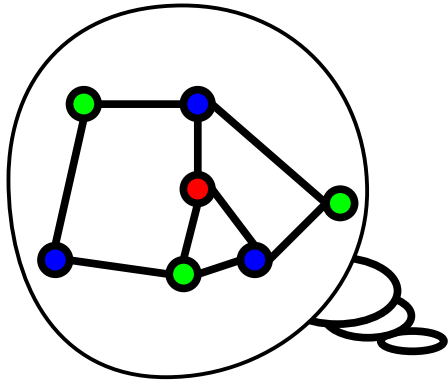
random $\pi: \{r,g,b\} \rightarrow \{r,g,b\}$
 for $I = 1, \dots, n$
 random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$



random j

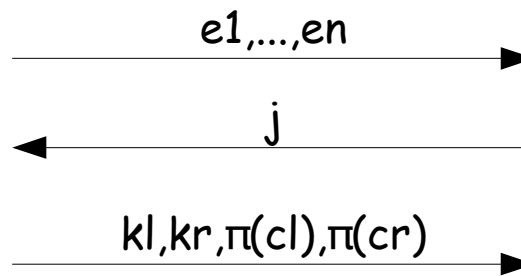
$e_l = \text{enc}_{k_l}(\pi(c_l)); e_r = \text{enc}_{k_r}(\pi(c_r))$
 $\pi(c_l), \pi(c_r) \in \{r,g,b\}; \pi(c_l) \neq \pi(c_r)$

Protocol



$n = 7$ nodes
 $m = 9$ edges

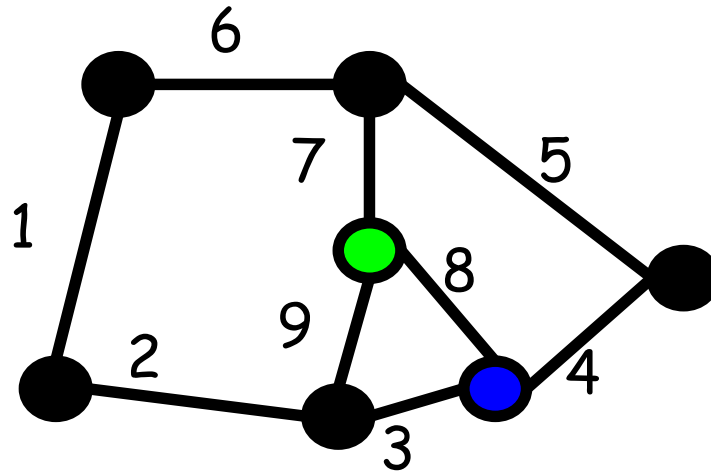
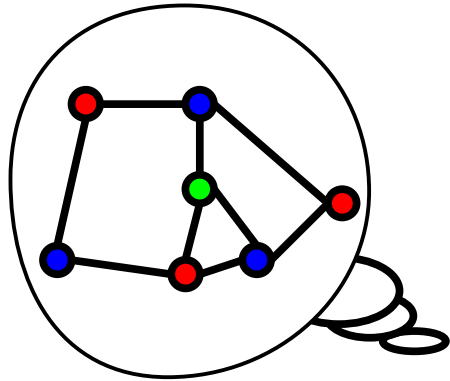
random $\pi: \{r,g,b\} \rightarrow \{r,g,b\}$
 for $I = 1, \dots, n$
 random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$



random j

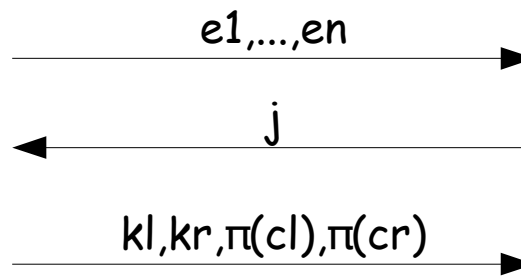
$e_l = \text{enc}_{k_l}(\pi(c_l)); e_r = \text{enc}_{k_r}(\pi(c_r))$
 $\pi(c_l), \pi(c_r) \in \{r,g,b\}; \pi(c_l) \neq \pi(c_r)$

Protocol



$n = 7$ nodes
 $m = 9$ edges

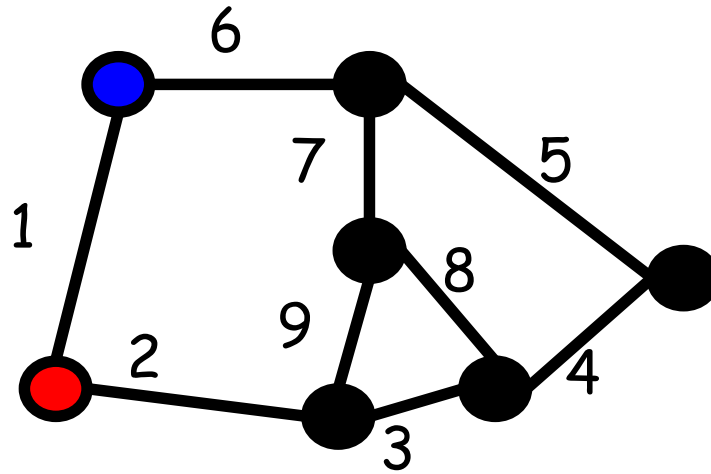
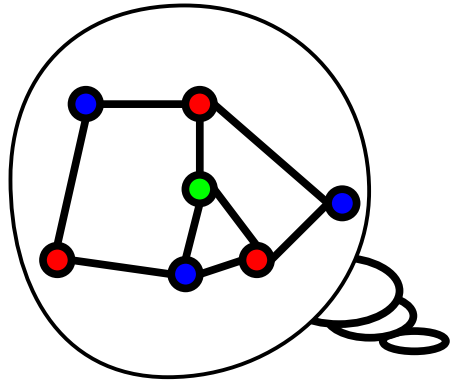
random $\pi: \{r,g,b\} \rightarrow \{r,g,b\}$
 for $I = 1, \dots, n$
 random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$



random j

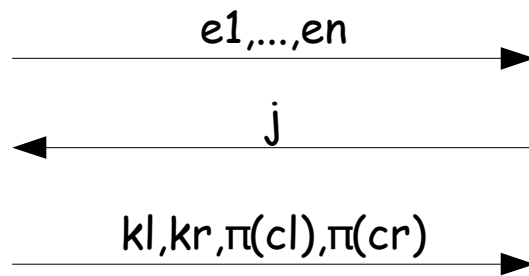
$e_l = \text{enc}_{k_l}(\pi(c_l)); e_r = \text{enc}_{k_r}(\pi(c_r))$
 $\pi(c_l), \pi(c_r) \in \{r,g,b\}; \pi(c_l) \neq \pi(c_r)$

Protocol



$n = 7$ nodes
 $m = 9$ edges

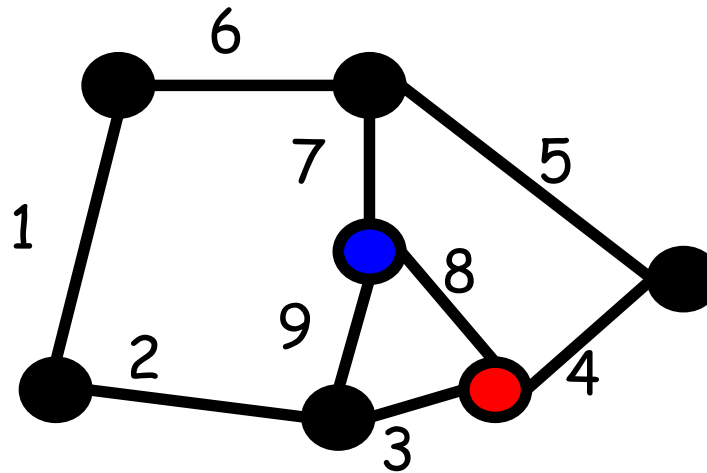
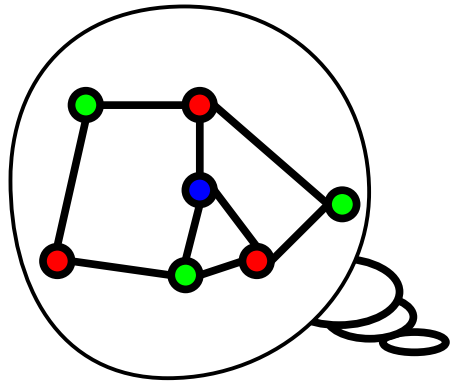
random $\pi: \{r,g,b\} \rightarrow \{r,g,b\}$
 for $I = 1, \dots, n$
 random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$



random j

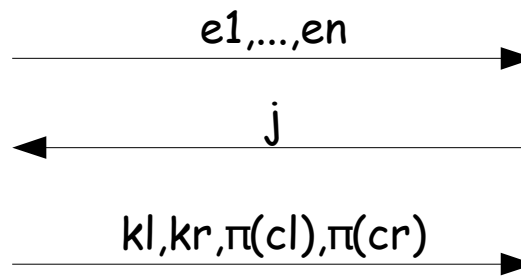
$e_l = \text{enc}_{k_l}(\pi(c_l)); e_r = \text{enc}_{k_r}(\pi(c_r))$
 $\pi(c_l), \pi(c_r) \in \{r,g,b\}; \pi(c_l) \neq \pi(c_r)$

Protocol



$n = 7$ nodes
 $m = 9$ edges

random $\pi: \{r,g,b\} \rightarrow \{r,g,b\}$
 for $I = 1, \dots, n$
 random k_i
 $e_i = \text{enc}_{k_i}(\pi(c_i))$



random j

$e_l = \text{enc}_{k_l}(\pi(c_l)); e_r = \text{enc}_{k_r}(\pi(c_r))$
 $\pi(c_l), \pi(c_r) \in \{r,g,b\}; \pi(c_l) \neq \pi(c_r)$

Nice looking protocol.....



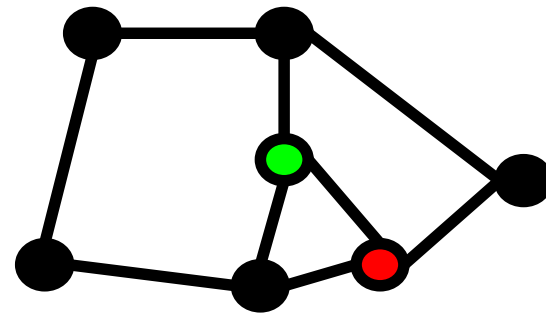
Does Bob really not learn anything?

Why should Bob believe Alice?



Zero-Knowledge ?

The verifier learns nothing: anything he outputs he can also output w/o prover.

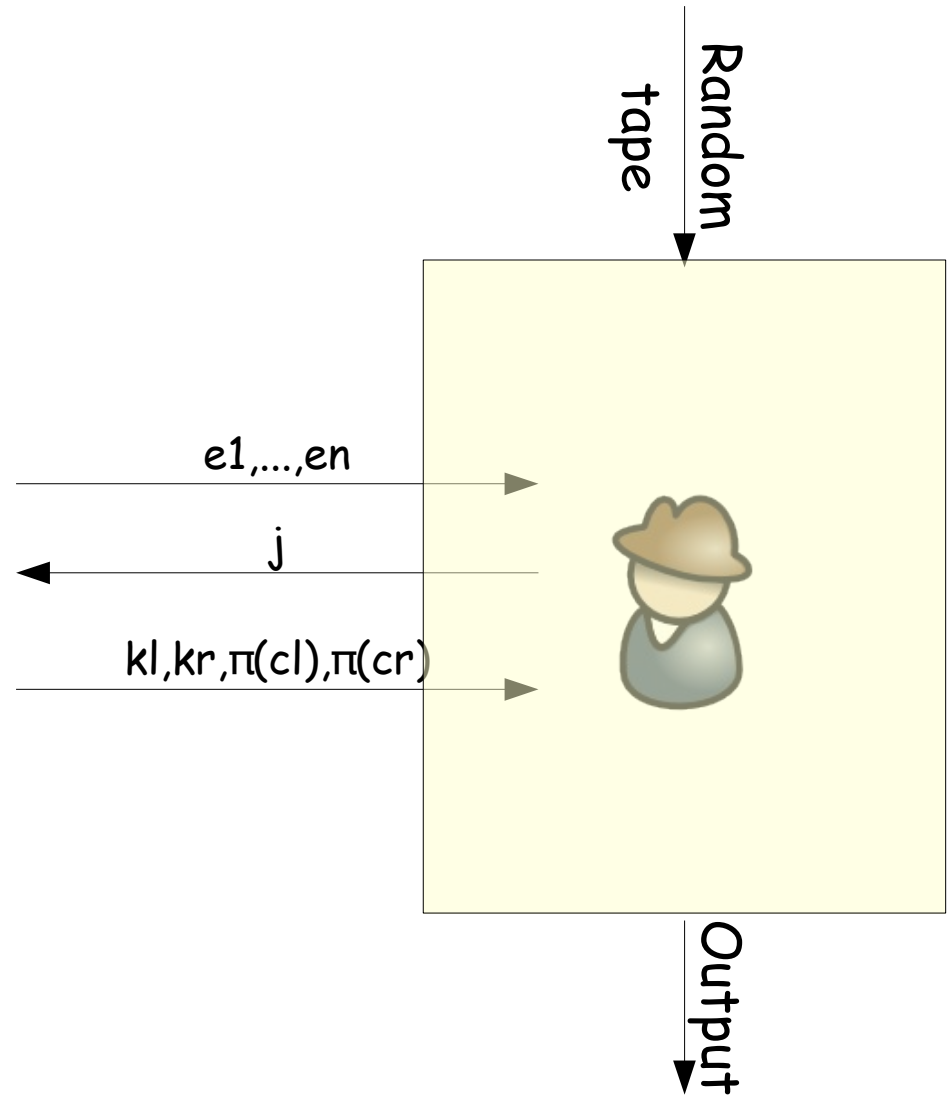


Claim: For every (arbitrarily behaving) verifier there exists a simulator that producing the same w/o interaction with prover.

Notice: verifier is probabilistic Turing machine, i.e., is deterministic with a random tape

Simulation

- Guess edge j'
- pick random color for cl
 cr (different)
- for $i=1,\dots,n$
 - pick ki
 - $e_i = \text{enc}_{ki}(0)$
except $e_r = \text{enc}_{kr}(cr)$
 $e_l = \text{enc}_{kl}(cl)$
- send e_1,\dots,e_n
- if guess incorrect, start over



Zero-Knowledge!

To prove claim: messages sent have same distribution.

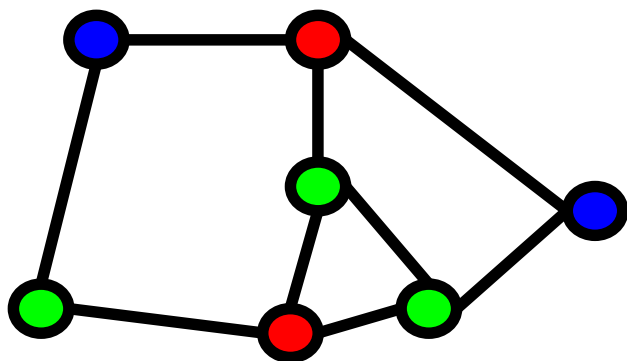
What's different:

- Picked color randomly: same as random perm.
- Encrypted 0's: cannot tell difference if it is a (semantically) secure encryption scheme.

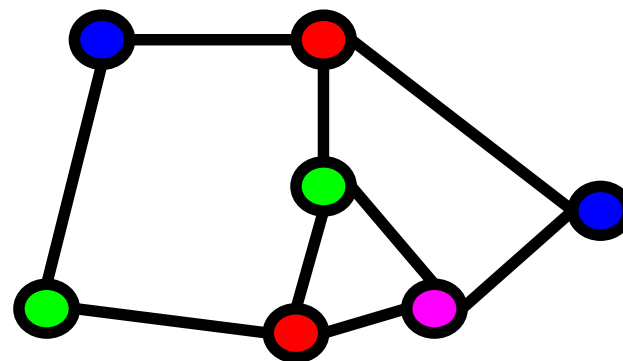
Input is the same, then so must be the output!

Soundness

What if graph was not 3-colorable (or Alice doesn't know)



Neighbors have same color



More than 3 colors

Probability to accept faulty graph:

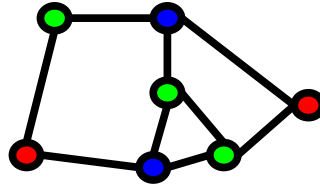
$1 - 1/m$ per round

Prob. Bob accepts all k rounds:

$$(1 - 1/m)^k = \text{exp. small in } k$$

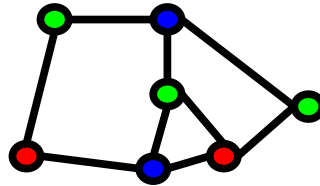
Proof of Knowledge

1. round: e_1, \dots, e_n



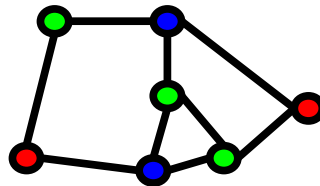
⋮

i -th. round: e_1, \dots, e_n



⋮

n^2 . round:
 e_1, \dots, e_n

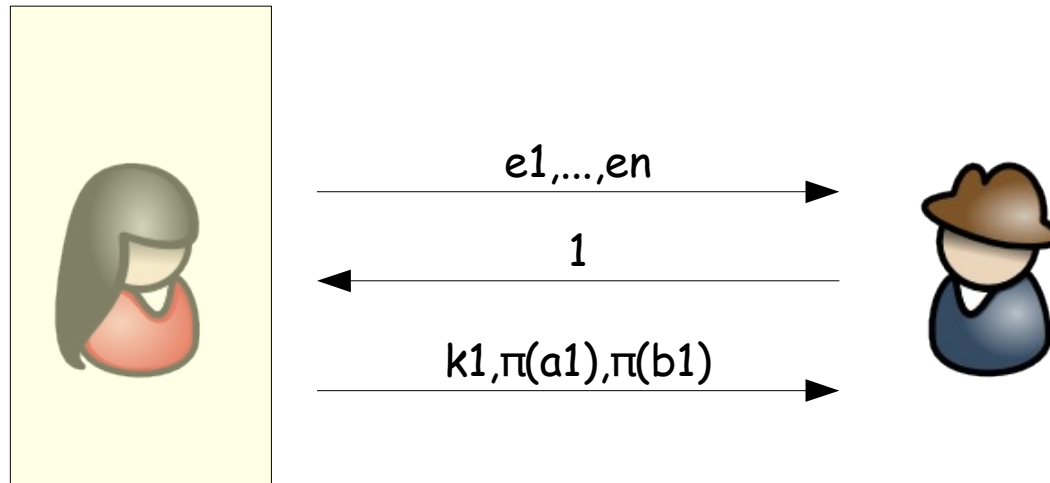
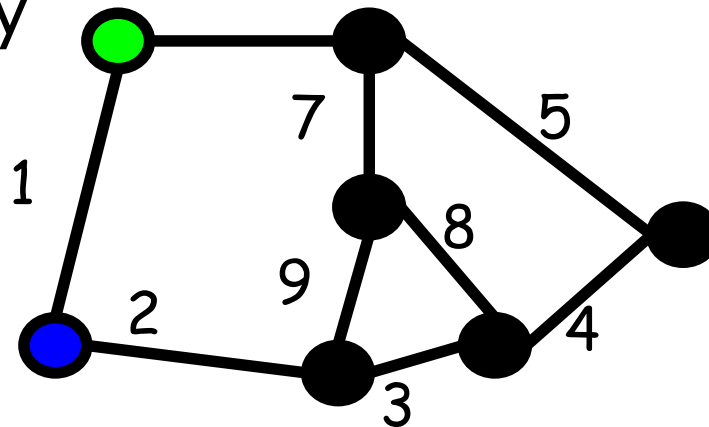


Assume verifier
accepts with

$$p > (1 - 1/m)^k$$

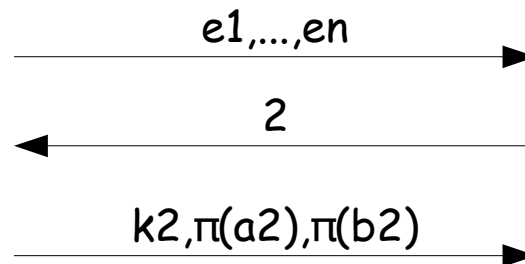
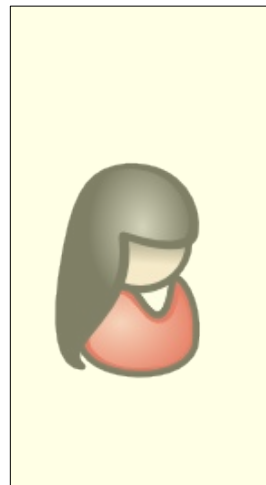
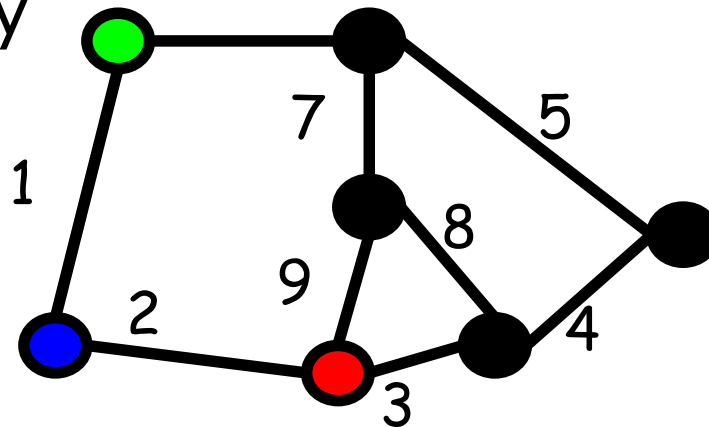
Extraction of Knowledge

Intuitively, if Bob accepts, with high probability at least one graph was ok



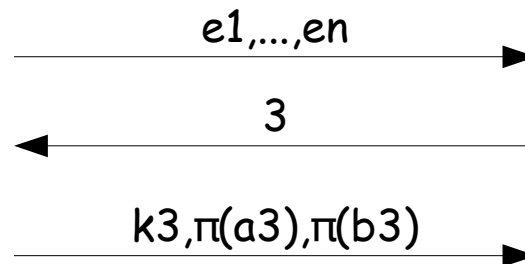
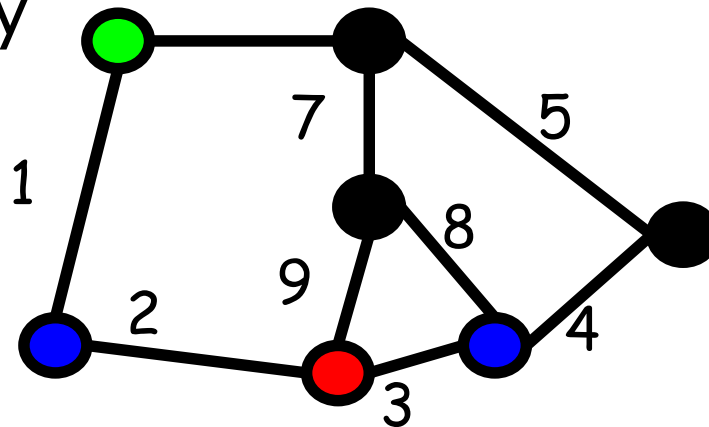
Extraction of Knowledge

Intuitively, if Bob accepts, with high probability at least one graph was ok



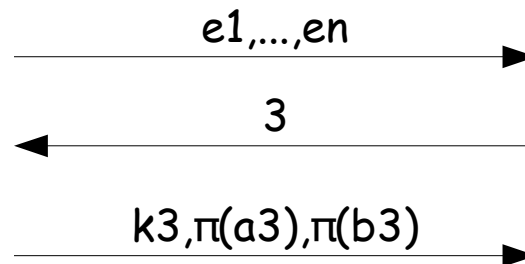
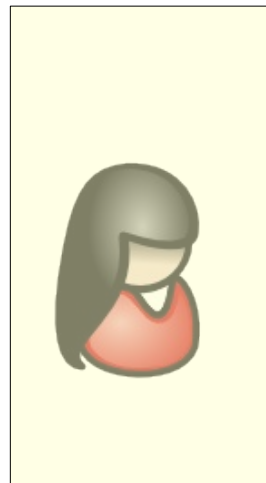
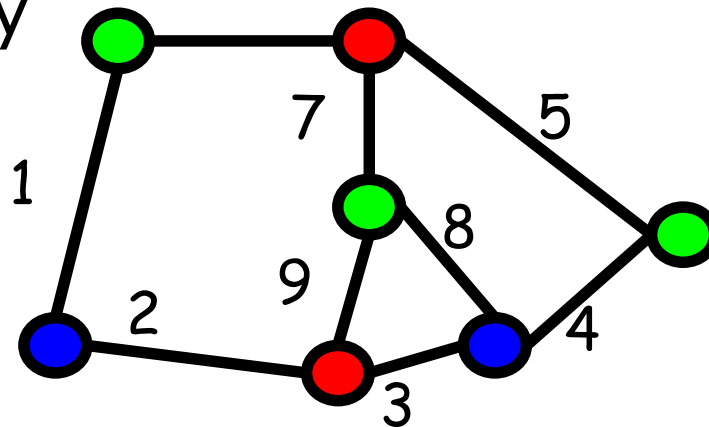
Extraction of Knowledge

Intuitively, if Bob accepts, with high probability at least one graph was ok



Extraction of Knowledge

Intuitively, if Bob accepts, with high probability at least one graph was ok



A Quick Number Theory Reminder

Number Theory: Discrete Logarithm

Group G

- $G = \langle g \rangle = \{1=g^0, g, g^2, \dots, g^{q-1}\}$
- q : order (so we are computing mod q in exponent)
- g : generator

Given x it is easy to compute

- $y = g^x, y = g^{1/x}, \dots$

Discrete Log Assumption:

- Given y it is hard to compute x s.t. $y = g^x$

Diffie-Hellman Assumption:

- Given g, y, h it is hard to compute $z = h^x$ where $x = \log_g y$

Decisional Diffie-Hellman Assumption

- Given g, y, h, z it is hard to decide if $\log_h z = \log_g y$

Number Theory: RSA & Factoring

Computing $n = pq$ with p and q primes is easy

Given n , finding p and q s.t. $n = pq$ is hard
(Factoring Assumption)

Consider group $Z_n^* = \{i : 0 < i < n, \gcd(i, n) = 1\}$

RSA: given n and e computing

- $y = g^e \pmod n$ is easy
- $y = g^{1/e} \pmod n$ is hard (RSA Assumption)

Strong RSA Assumption: given n and z finding

- u, e s.t. $z = u^e \pmod n$ is hard

Technologies

Efficient Zero-Knowledge Proofs

Zero Knowledge Proofs

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

Prover wants to convince verifier that she *knows* $x = \log_g y$ such that verifier only learns y and g .



Prover:

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

Verifier:



random r

$$t := g^r$$

t



c



$$s := r - cx$$

s



random c

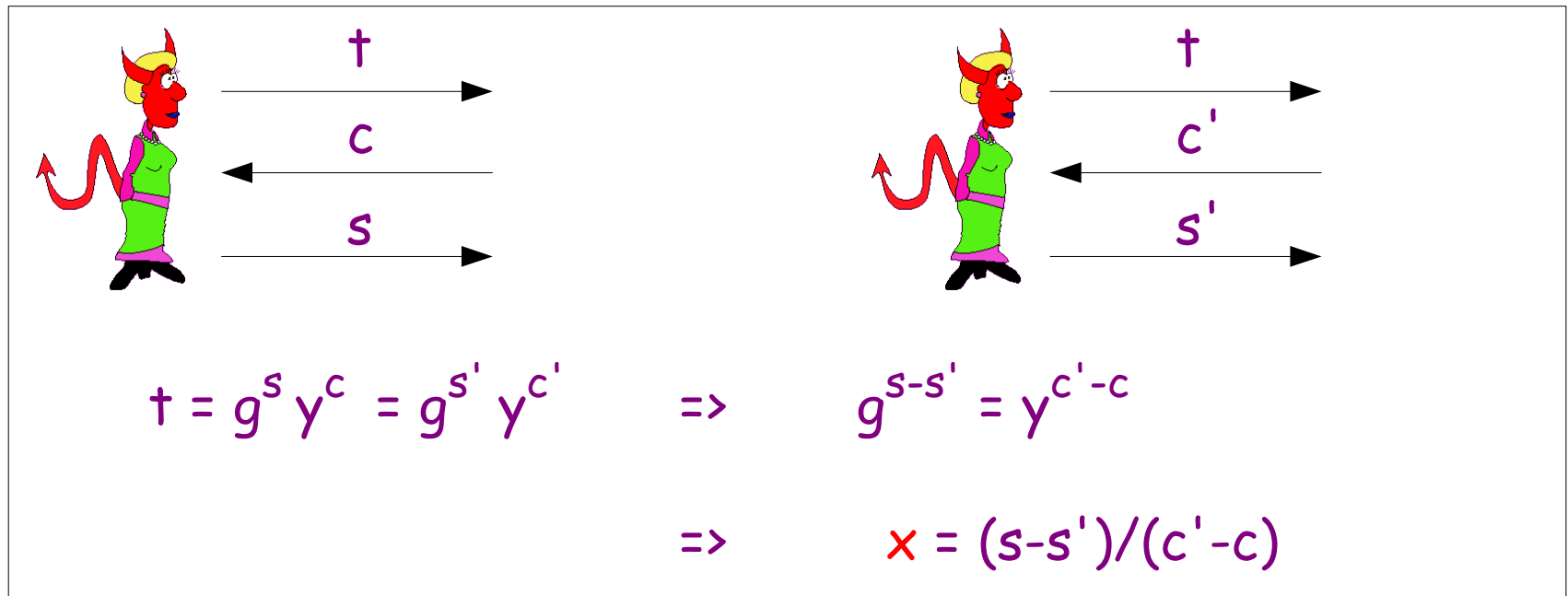
$$t = g^s y^c$$

Zero Knowledge Proofs: Security

Proof of Knowledge Property:

If prover is successful, then she “knows” $x = \log_g \gamma$, i.e., one can extract x from her.

Idea: run two copies of Alice.



(One would need to consider success probabilities.....)

Zero Knowledge Proofs: Security

Zero-knowledge property:

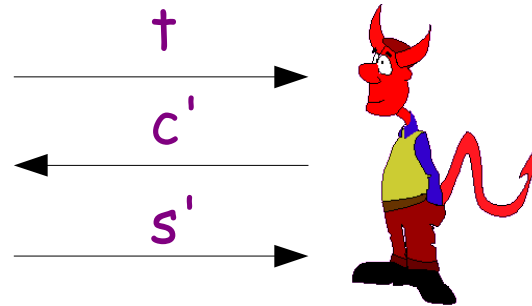
If verifier does not learn anything (except the fact that Alice knows $x = \log_g y$)

Idea: One can simulate whatever Bob "sees".

Choose random c, s

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

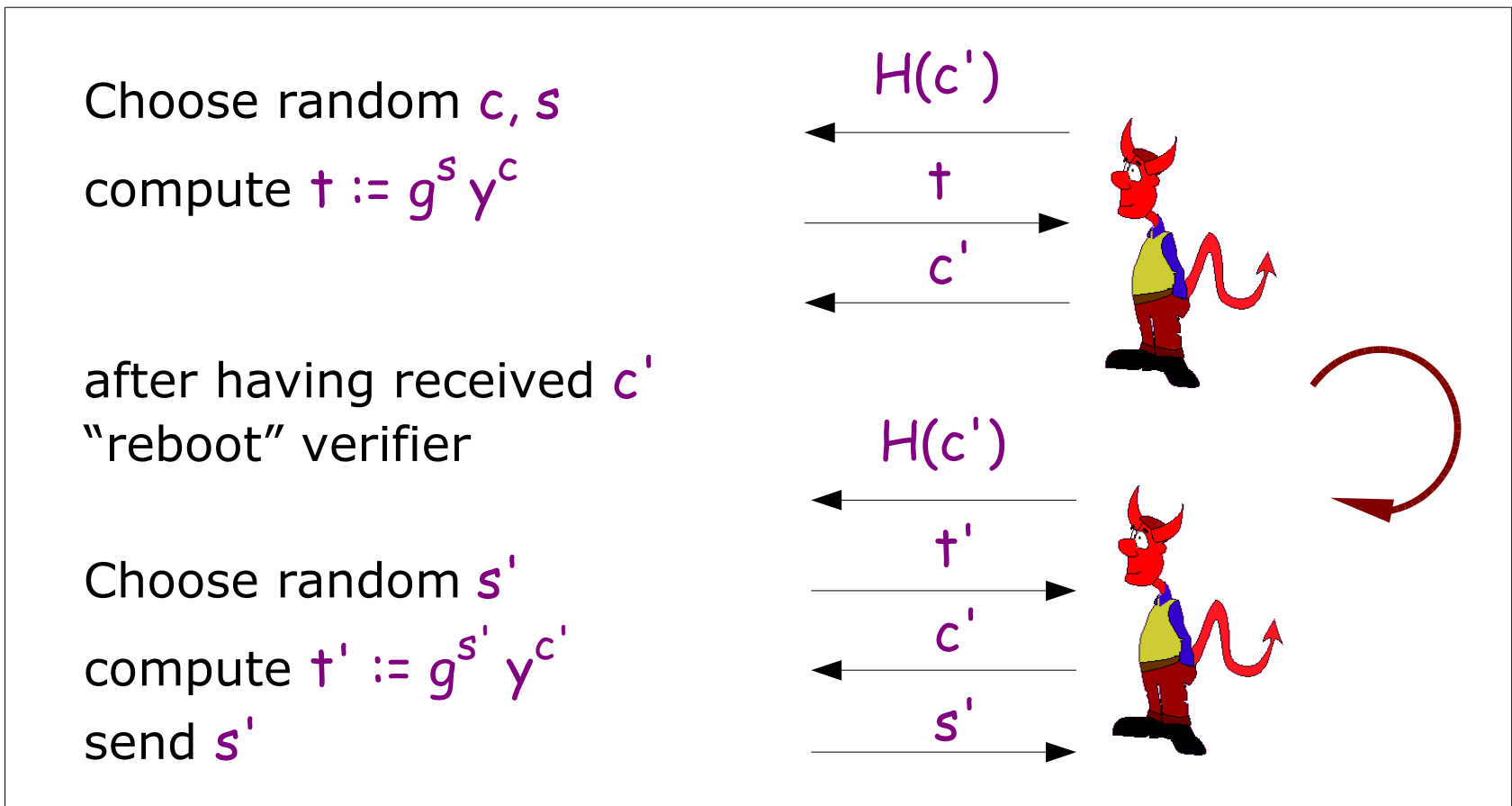
if $c = c'$ send $s' = s$,
otherwise restart



(works only if c' comes from small domain, but can be overcome by various means.)

Zero Knowledge Proofs: Security

One way to get large c :



(One would need to consider success probabilities.....)

Zero Knowledge Proofs

Non-interactive (Fiat-Shamir heuristic):

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

Logical combinations:

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

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

Intervals and groups of different order (under SRSA):

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

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

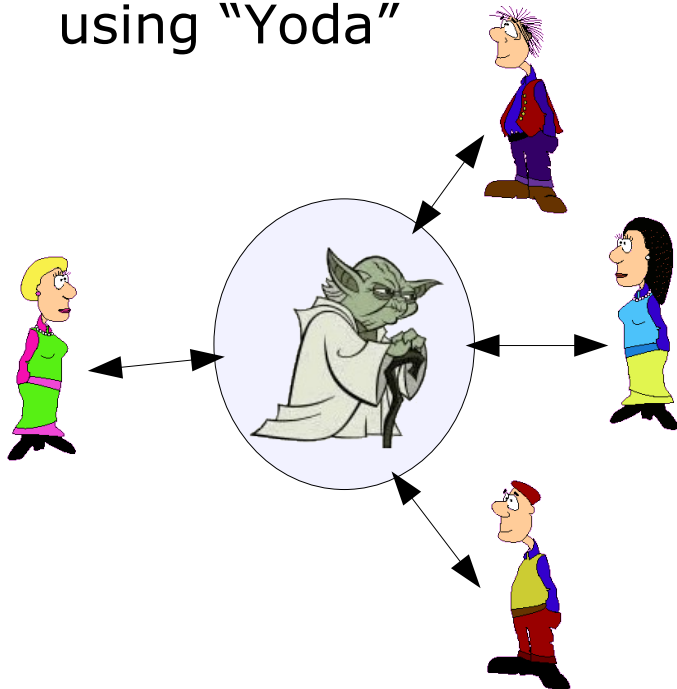
Referenes

- 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>

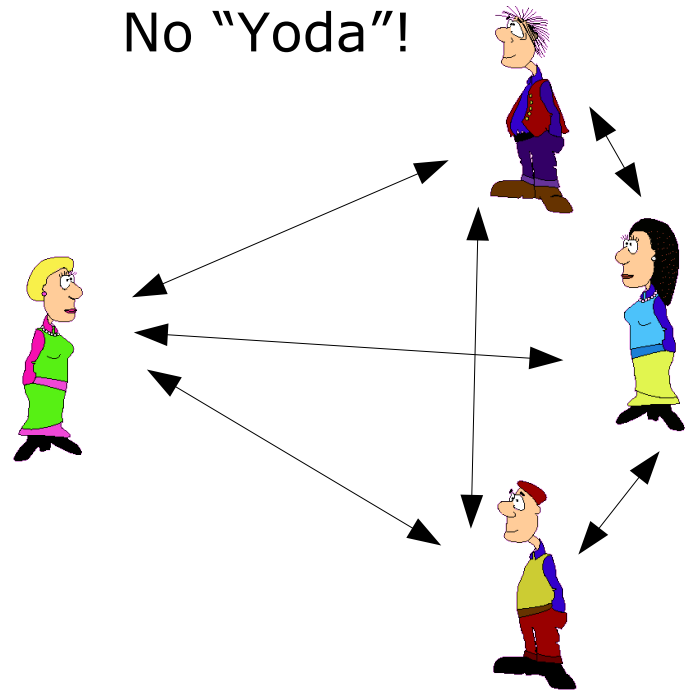
Security of Anonymous Credentials

Approach:

Ideal World
using "Yoda"



Real World Protocol
No "Yoda"!

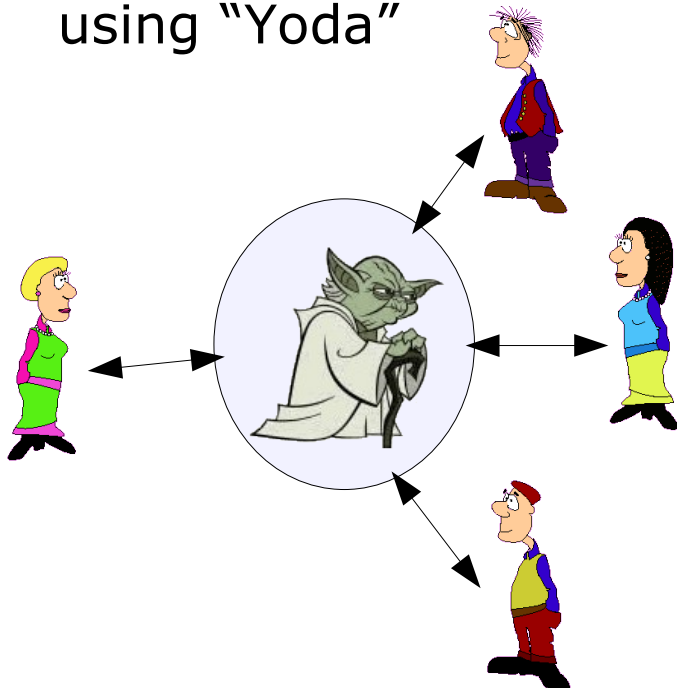


1. Specify how "Yoda" should behave
2. Prove that real protocol achieves the same

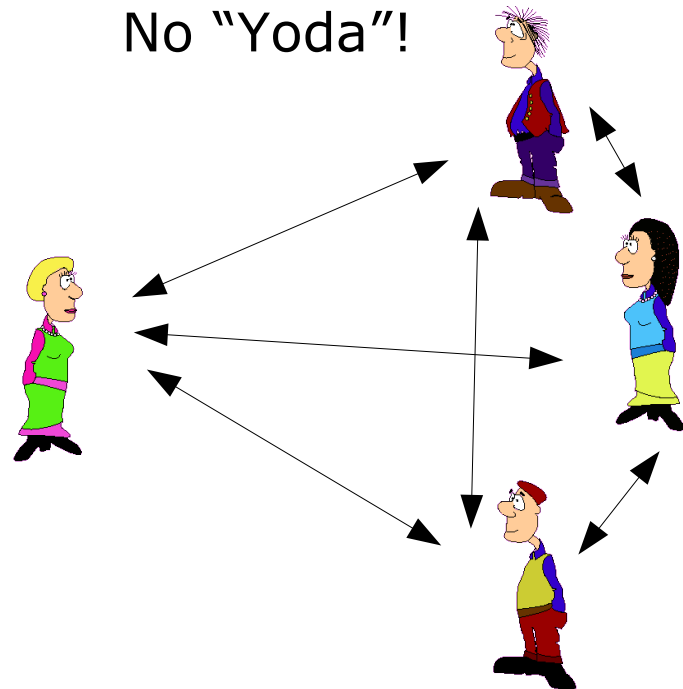
Security of Anonymous Credentials

Approach:

Ideal World
using "Yoda"



Real World Protocol
No "Yoda"!

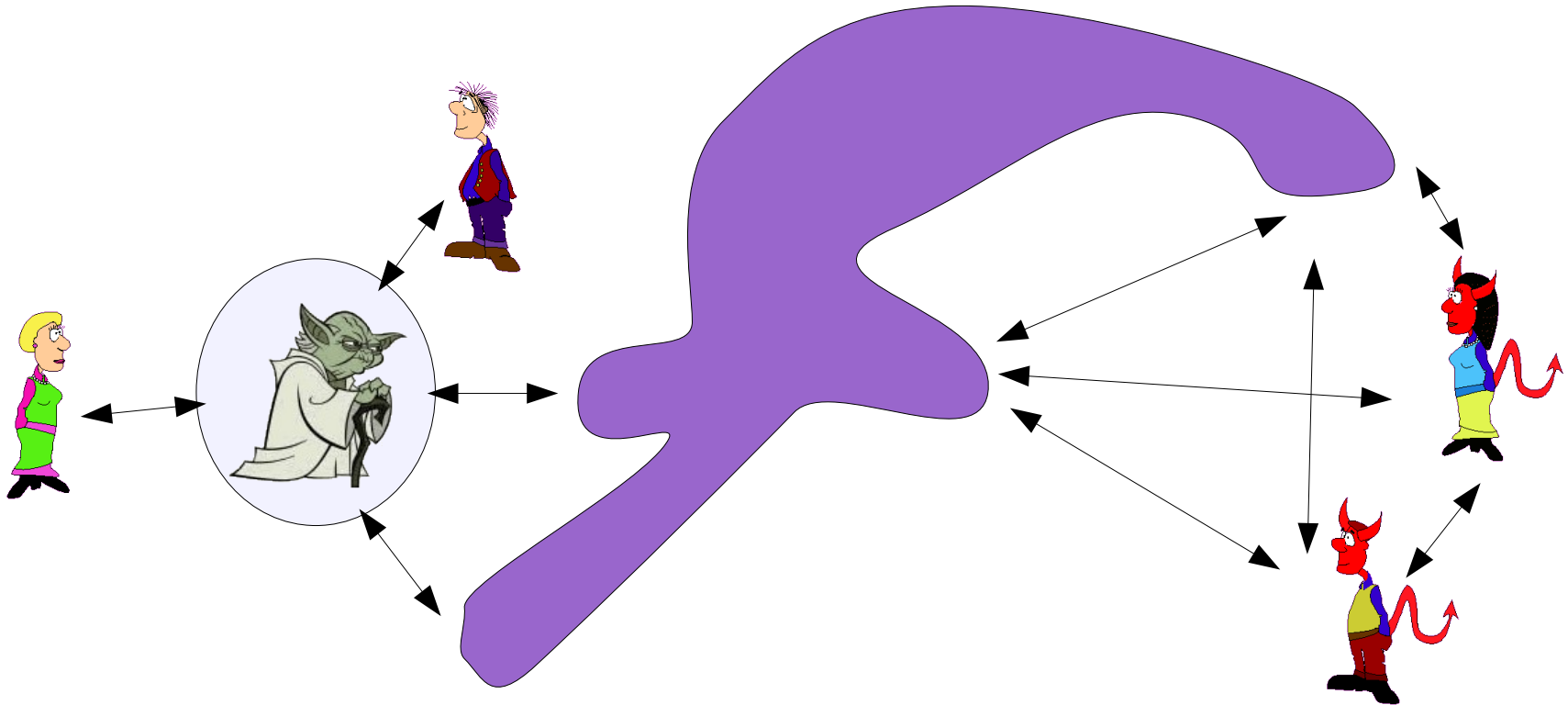


1. Specify how "Yoda" should behave
2. Prove that real protocol achieves the same

Adversary learns not more in Real World than in Ideal World

Security of Anonymous Credentials

Proof Method:



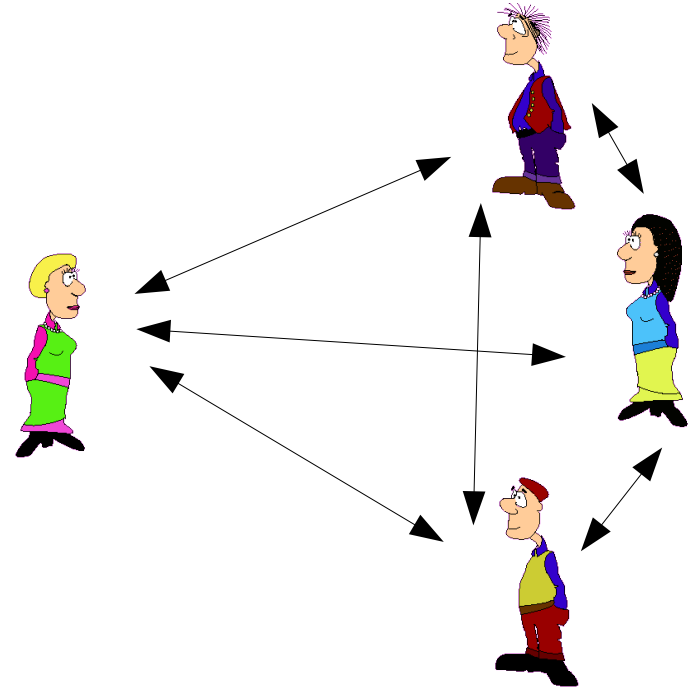
Real World Adversary can't tell the difference between interacting with the real player and the blue simulator

System Behavior of Anonymous Credentials

Players: Users, Orgs, Revs, CA

Operations:

- Register(User(), CA())
- FormNym(User(), Org())
- GetCred(User(), Org())
- VerifyCred(User(), Org())
- VerifyCredOnNym(User(), Org())
- AnonRev(Org, Rev)
- LocalAnonRev(Org, Rev)



System Behavior of Anonymous Credentials

- Register(User(), CA())
 - Yoda sends ID_U to CA.
 - Yoda remembers ID_U as registered.
- FormNym(User(), Org())
 - Yoda checks whether ID_U is registered.
 - Yoda generates Nym N .
 - Yoda sends FromNym: N to Org.
 - If Org agrees, Yoda sends N to User and stores (Nym, ID_U , N , ID_Org).
- GetCred(User(), Org())
 - User sends N , ID_Org to Yoda
 - Yoda checks if he has stored (Nym, ID_U , N , ID_Org).
 - Yoda sends Cred: N to Org.
 - If Org agrees, Yoda sends Cred:OK to User and stores (Cred, ID_U , N , ID_Org).

System Behavior of Anonymous Credentials

- VerifyCred(User(), OrgI(), OrgV())
 - *User sends N_I, ID_OrgI, ID_OrgV to Yoda*
 - *Yoda checks if he has stored $(Cred, ID_U, N_I, ID_OrgI)$.*
 - *If so, Yoda sends generates session id S and sends $Ver:(S, ID_OrgI)$ to OrgV and stores $(Ver, S, ID_U, N_I, ID_OrgI)$.*
- VerifyCredOnNym(User(), OrgI(), OrgV())
 - *User sends $N_I, ID_OrgI, ID_OrgV, N_V$ to Yoda*
 - *Yoda checks if he has stored $(Cred, ID_U, N_I, ID_OrgI)$.*
 - *Yoda checks if he has stored $(Nym, ID_U, N_V, ID_OrgV)$.*
 - *If so, Yoda sends generates session id S and sends $VerNym:(S, ID_OrgI, N_V)$ to OrgV and stores $(Ver, S, ID_U, N_I, ID_OrgI)$.*

System Behavior of Anonymous Credentials

- AnonRev(Org,Rev)
 - Org sends S to Yoda
 - Yoda checks whether it has stored $(Ver, S, ID_U, N_I, ID_OrgI)$.
 - If so, Yoda sends S to Rev whether it agrees.
 - If so, Yoda sends ID_U to Org
- LocalAnonRev(Org,Rev)
 - Org sends S to Yoda
 - Yoda checks whether it has stored $(Ver, S, ID_U, N_I, ID_OrgI)$.
 - If so, Yoda sends S to Rev whether it agrees.
 - If so, Yoda sends N_i to Org

Security Properties

By inspection...

- Anonymity and privacy is guaranteed
 - Org ever only see a user's pseudonym
 - Yoda tells them whether a credential is valid
 - Different transaction are not linkable
- Unforgeability enforces by Yoda
- Anonymity Revocation also done by Yoda
- Sharing prevented by Yoda
- Attributes can be added trivially
- Other extensions as well

Proof of Security for Signature Based Solution

- Left as exercise :-)
- Basic Idea:
 - FormNym(User(), Org())
 - Yoda checks whether ID_U is registered.
 - Yoda generates Nym N .
 - Yoda sends FromNym: N to Org.
 - If Org agrees, Yoda sends N to User and stores (Nym, ID_U, N, ID_Org) .
 - Assume User is trusted Org is not
 - Generate fresh user's secret key
 - Generate public key/ pseudonym and send to Org
 - If org accepts protocol, tell yoda that we agree