Identification and Zero-Knowledge Proof

Outline

- Introduction to ID schemes
- Password-based ID schemes
- Zero-knowledge proof systems
- Public-key based ID schemes
  - Schnorr ID scheme

Identification

- How to identify yourself over the Internet
  - E.g., remotely login a server
- A naïve approach:

Client $\xrightarrow{\textit{username, pw}}$ Server

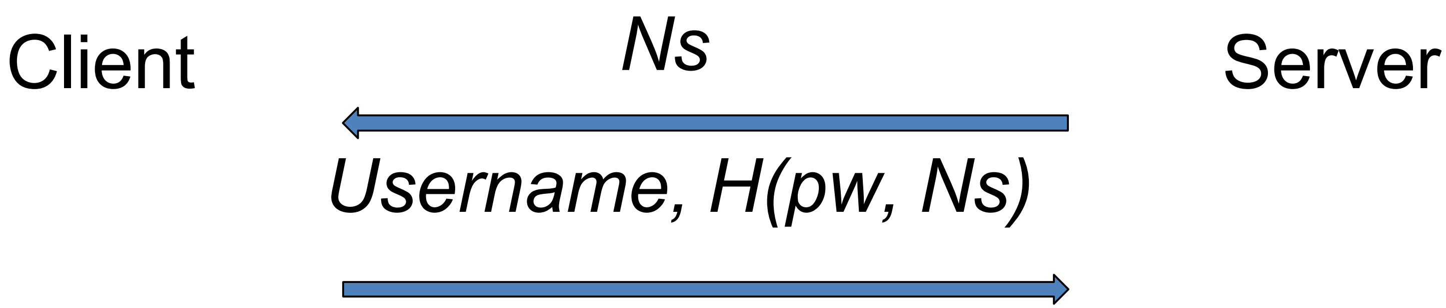
Improve Scheme I

Client *username, H(pw)* Server

$$\text{Client} \xrightarrow{\textit{username, H(pw)}} \text{Server}$$

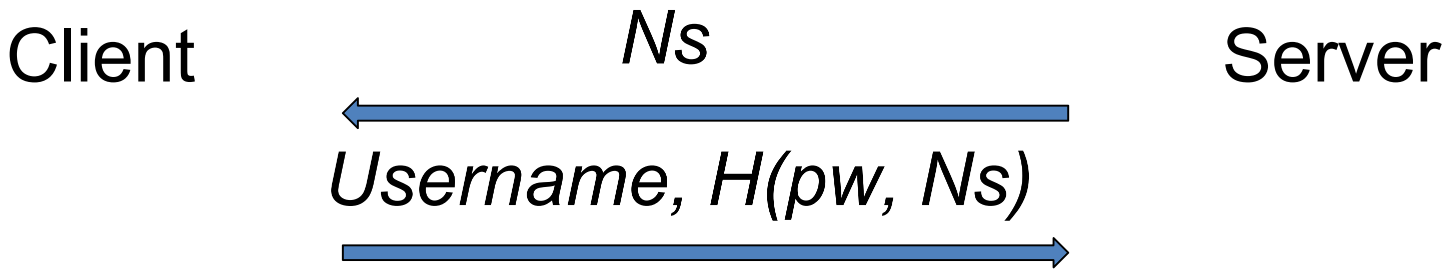
- Use a transformed password, e.g., H(pw) where H denotes a cryptographic hash function
- Is this approach secure?
  - No. A replay attack can still work
  - Rainbow table attack

Improved Scheme II



- The server sends a nonce Ns to the client as a challenge
  - Similar as a salt value in Unix
- The client gives a ***fresh*** response based on pw and Ns in each session

Improved Scheme II



- Is there any security issue here?

Problems with Password-based Identification Schemes

- Server has to store a password file
  - The password may be stored in a transformed form (e.g., H(pw))
  - Subject to brute-force and rainbow table attacks if the password file is leaked
- Client has to use different passwords for different sites
- This motivates us to use non-password based approaches

Public-key Identification Schemes

- Idea: the client proves to the server s/he has knowledge of a secret key corresponding to a public key
  - The public key is certified in the form of a digital certificate
  - Anyone can bind the username with the public key by verifying the certificate
- Question: how to ensure the secret key is not leaked in the identification process
  - The secret key should not be leaked even to the verifier!!

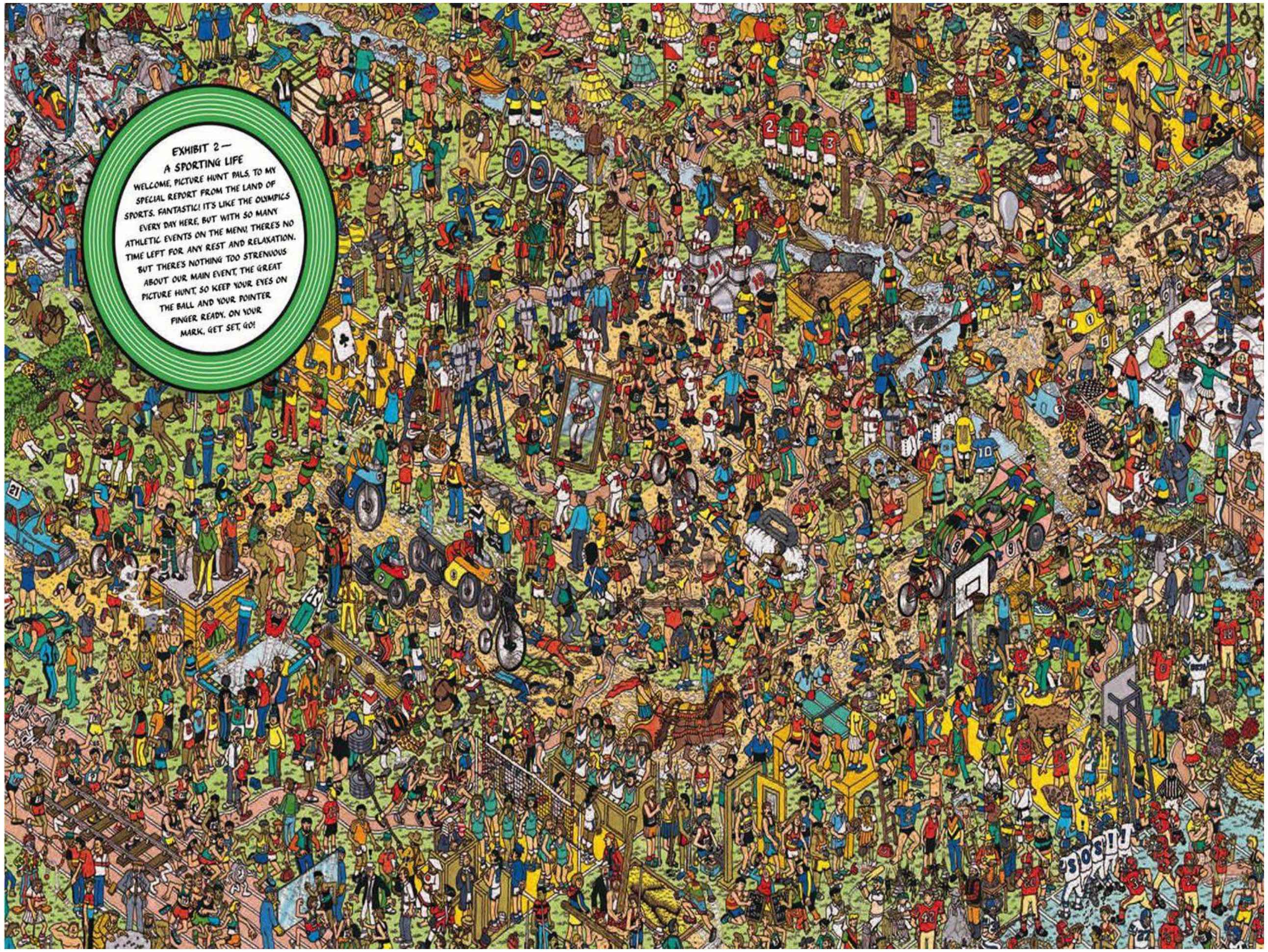
Zero Knowledge Proofs

- A protocol involving a prover and a verifier that enables the prover to convince a verifier that a statement is true without revealing any other information
  - Proving that one knows p,q such that n=pq
  - Proving that one knows x such $y = g^x \bmod p$
  - Proving that $y = Enc(pk, x)$ and $1 \le x \le 10$

Properties of Zero-Knowledge Proof

- Completeness
  - Given honest prover and honest verifier, the protocol succeeds with overwhelming probability
- Soundness
  - If the statement is wrong, the verifier rejects the proof with overwhelming probability
- Zero knowledge
  - the proof does not leak any additional information

Finding Wally

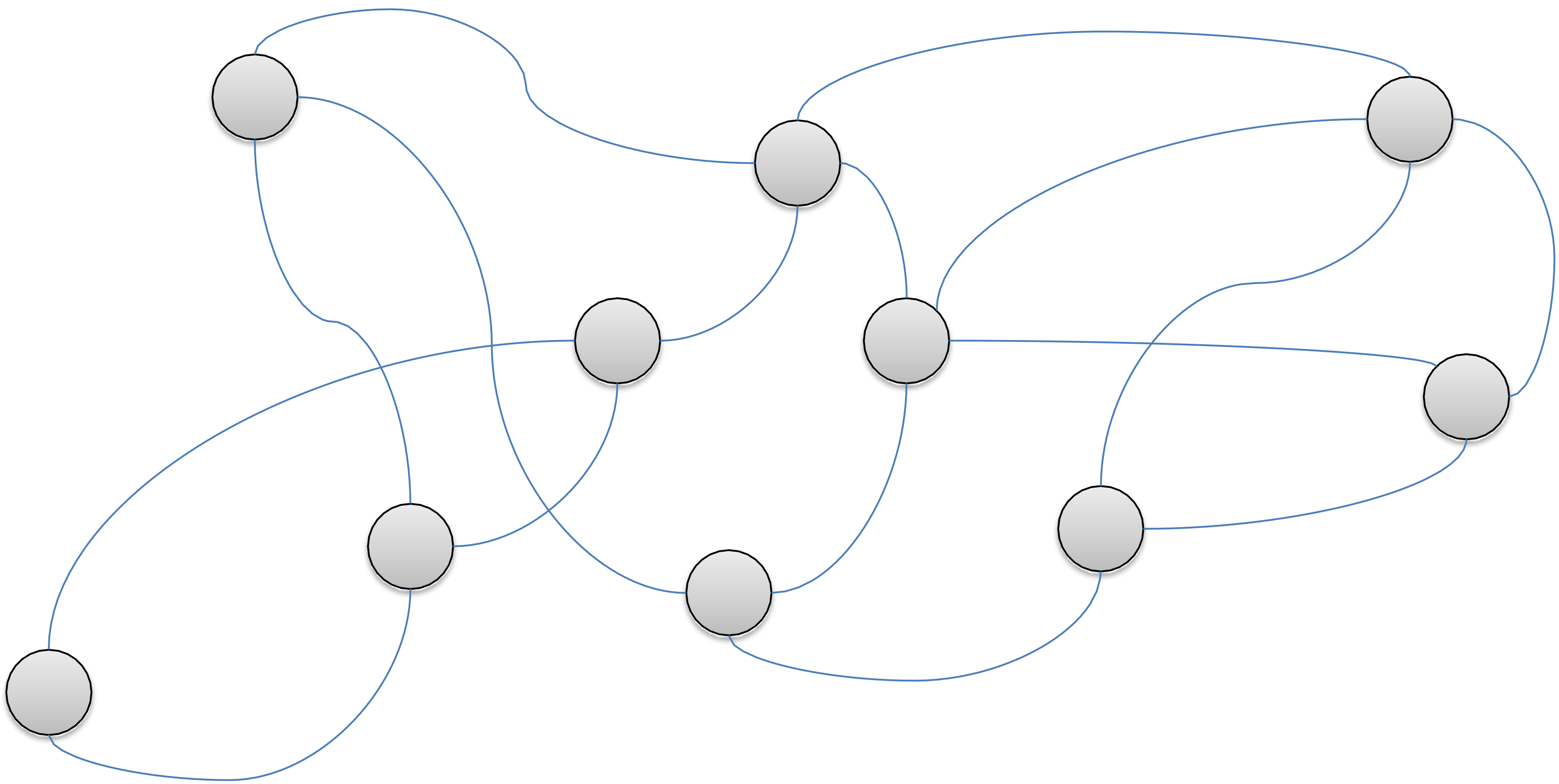


Actually…

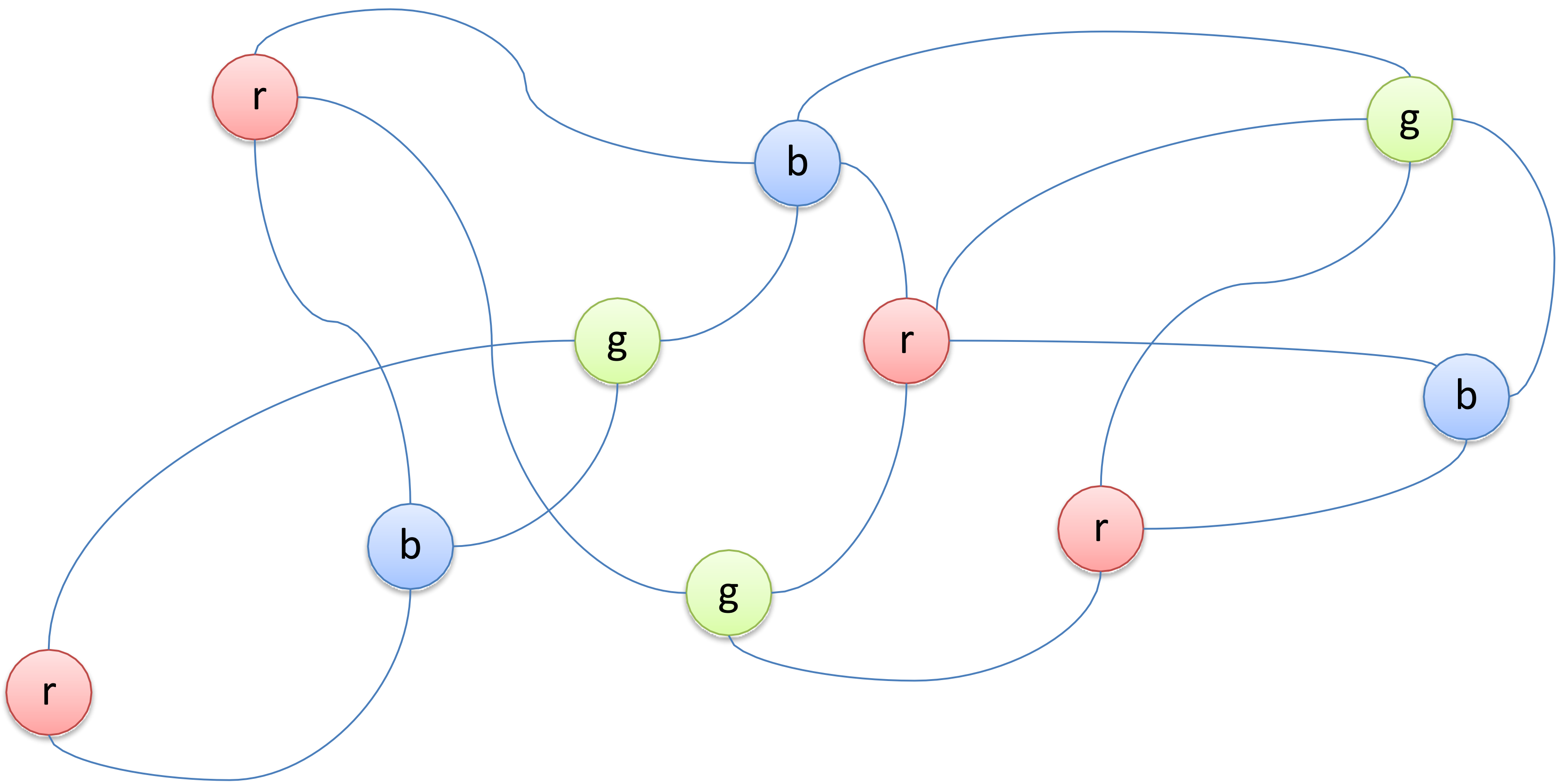
- …I'm not really interested in how to find him…
- …how can we convince someone **we** know where Wally is, without telling **them** where Wally is?

Another Problem

Label the following nodes so that two nodes sharing an edge will not have the same label, say, {r, g, b}



Okay… One possible answer

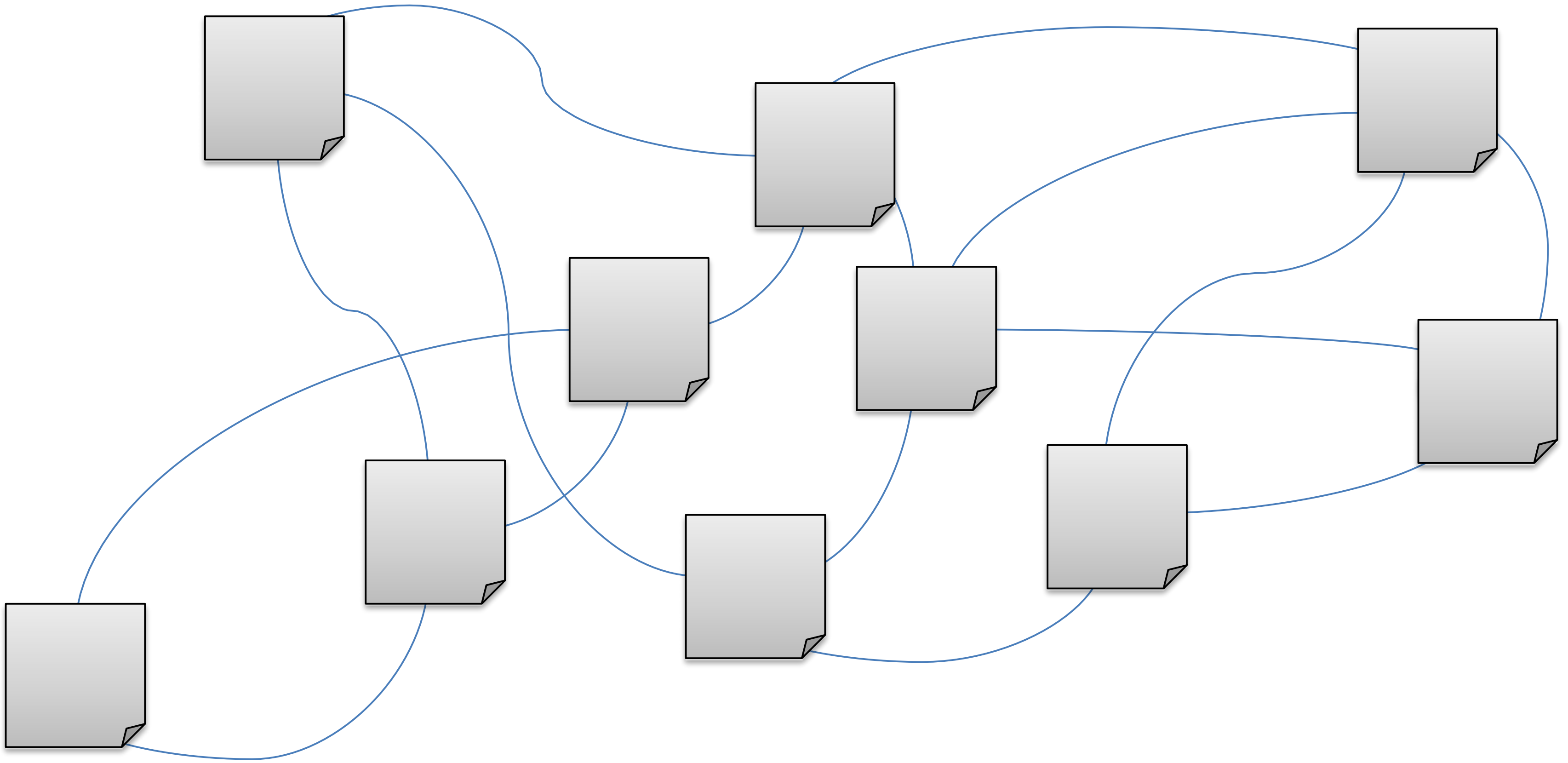


Zero-Knowledge proof

- Can I prove to you that I know the answer without showing you the answer???

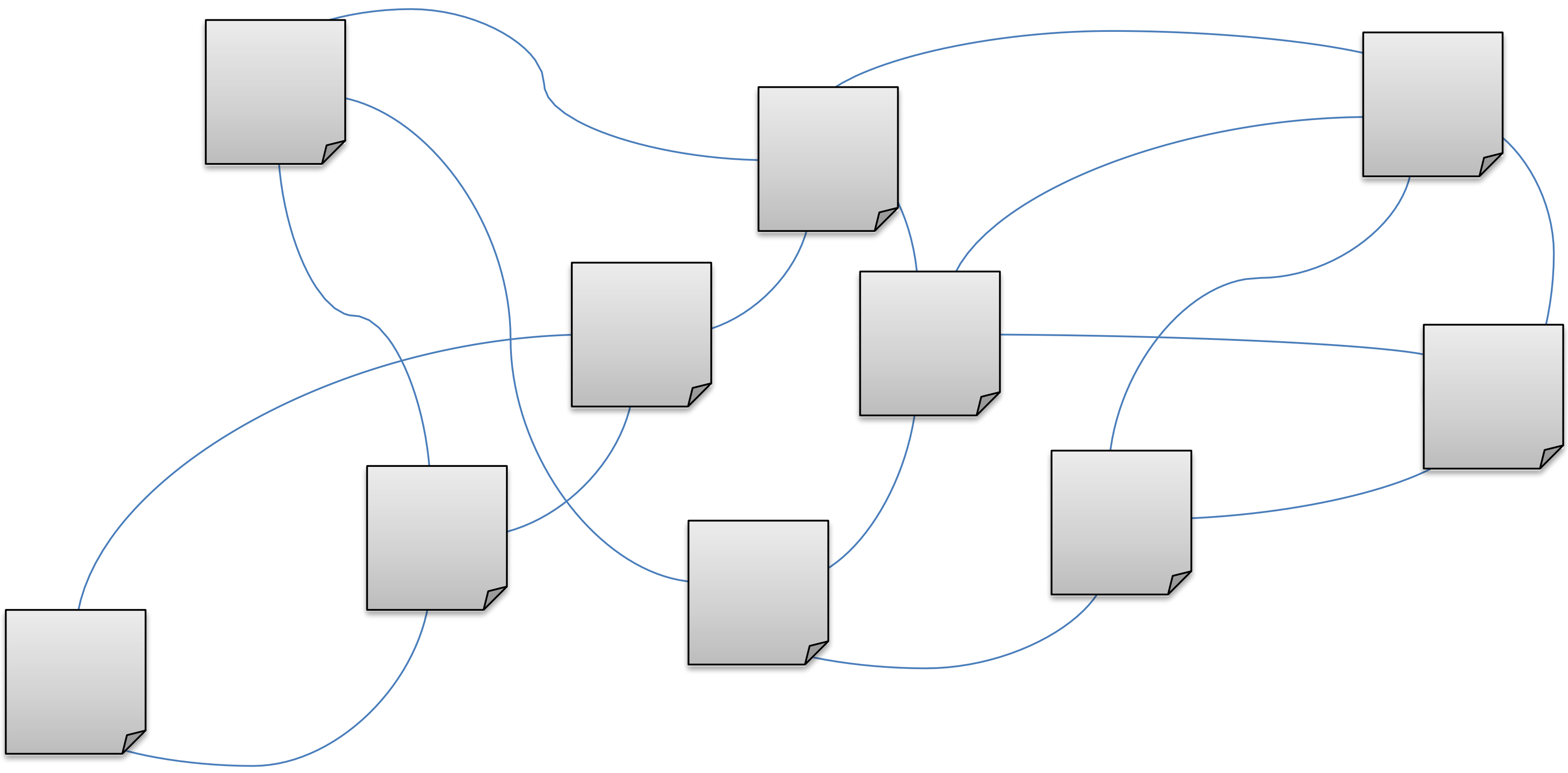
How can it be done?

Mask my answer



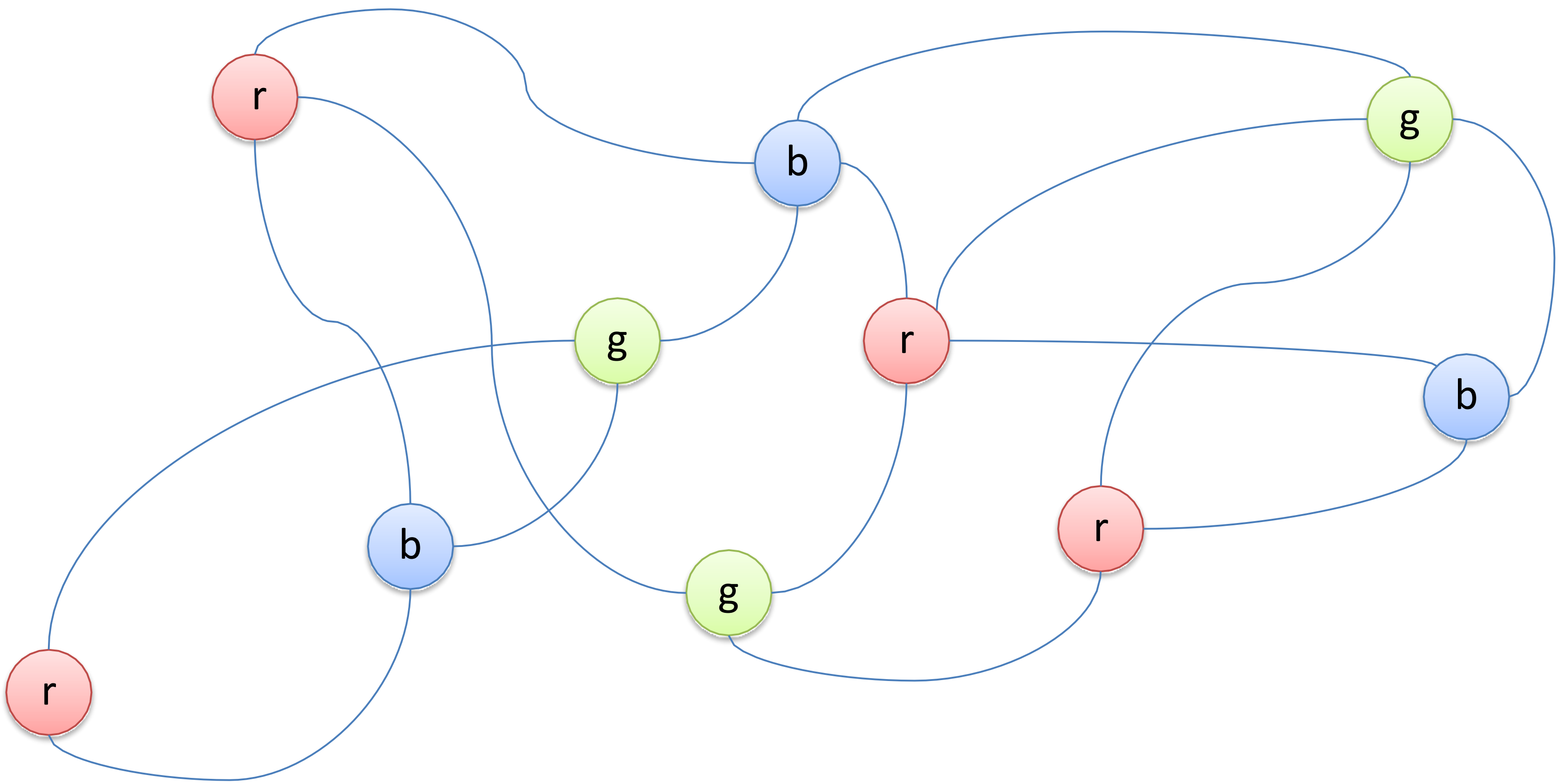
How can it be done?

Allow the verifier to open two connecting masks



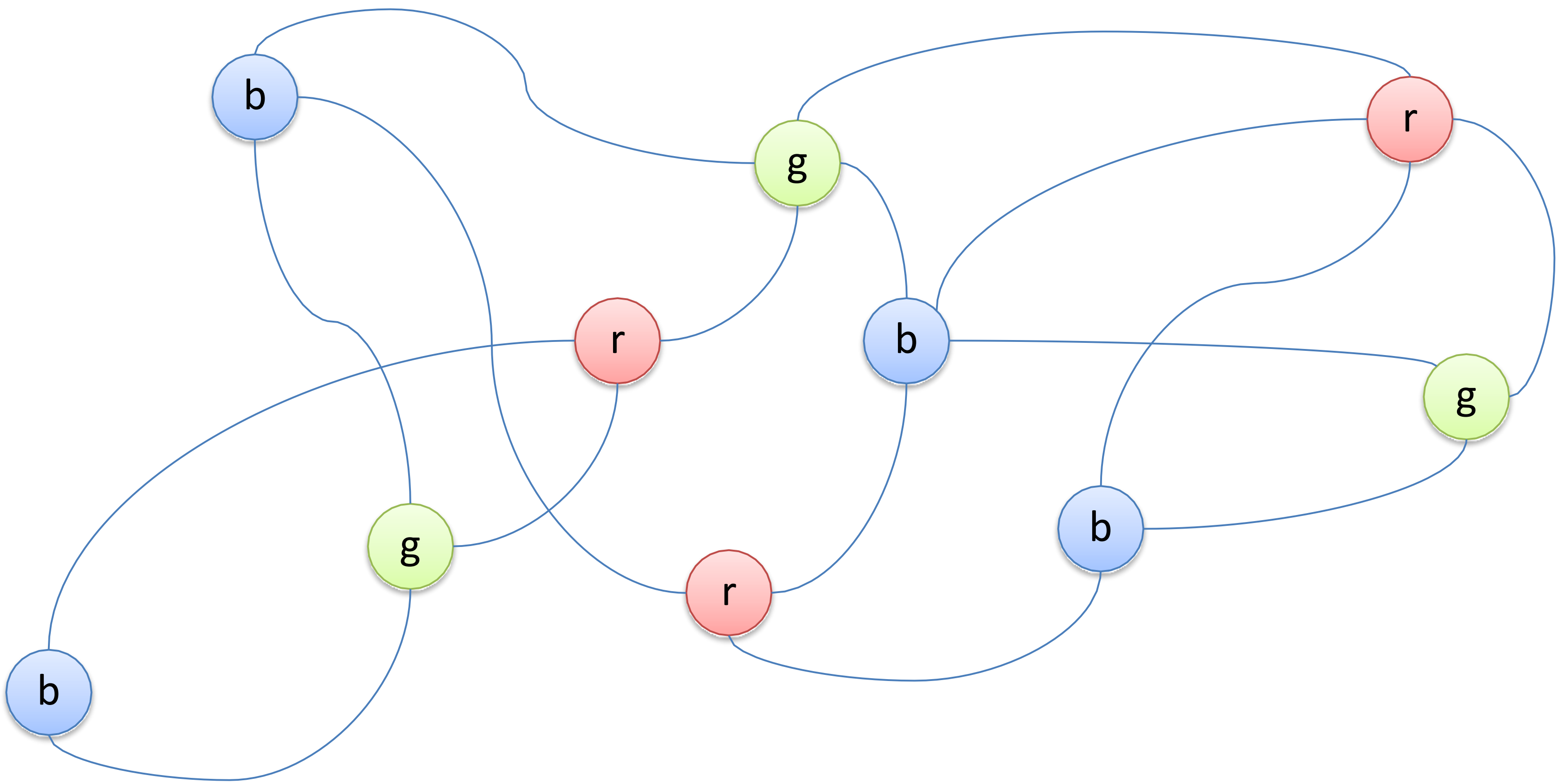
How can it be done?

Randomise my answer



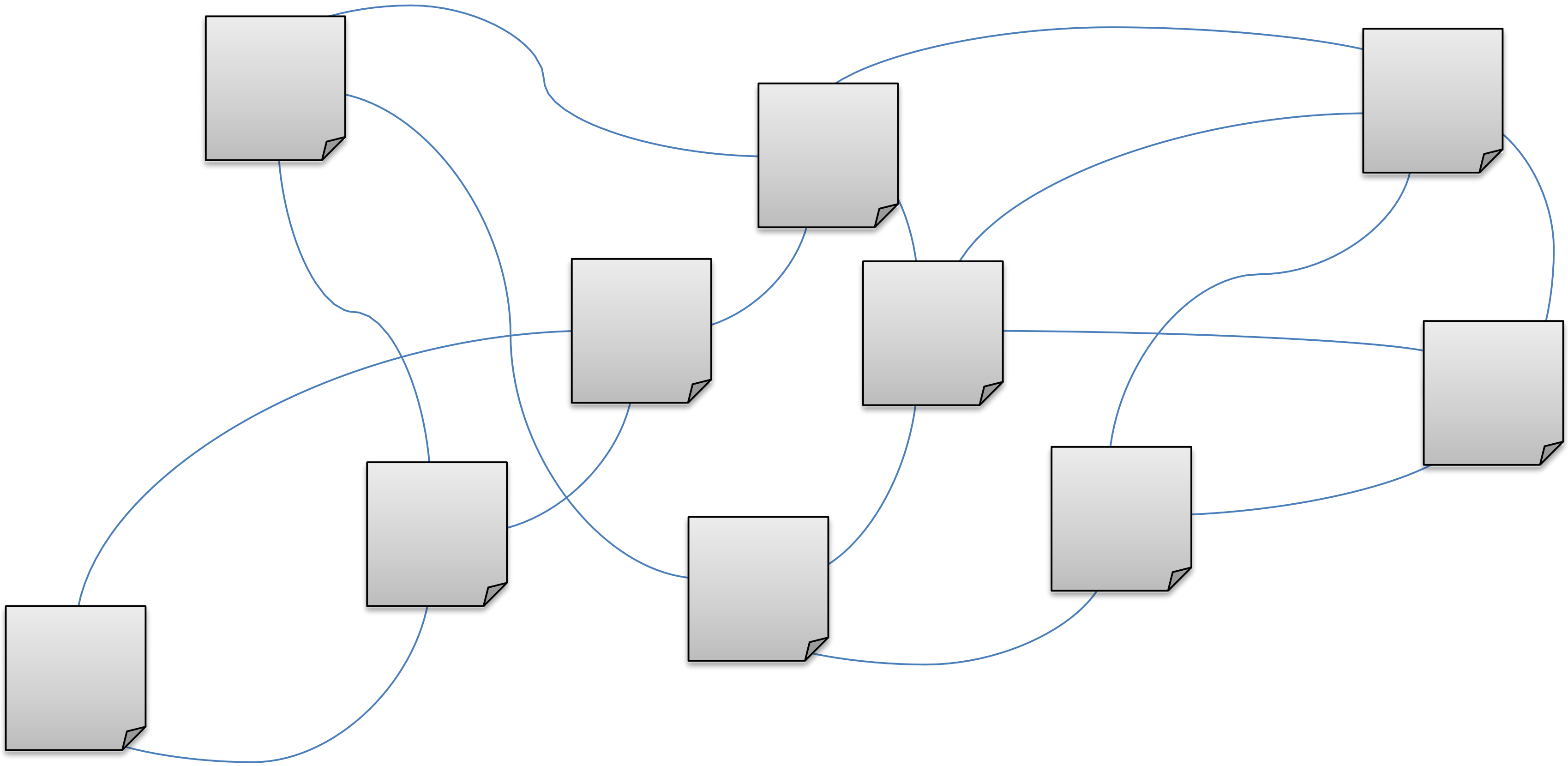
How can it be done?

Randomise my answer (in this case, r->b, b->g, g->r)



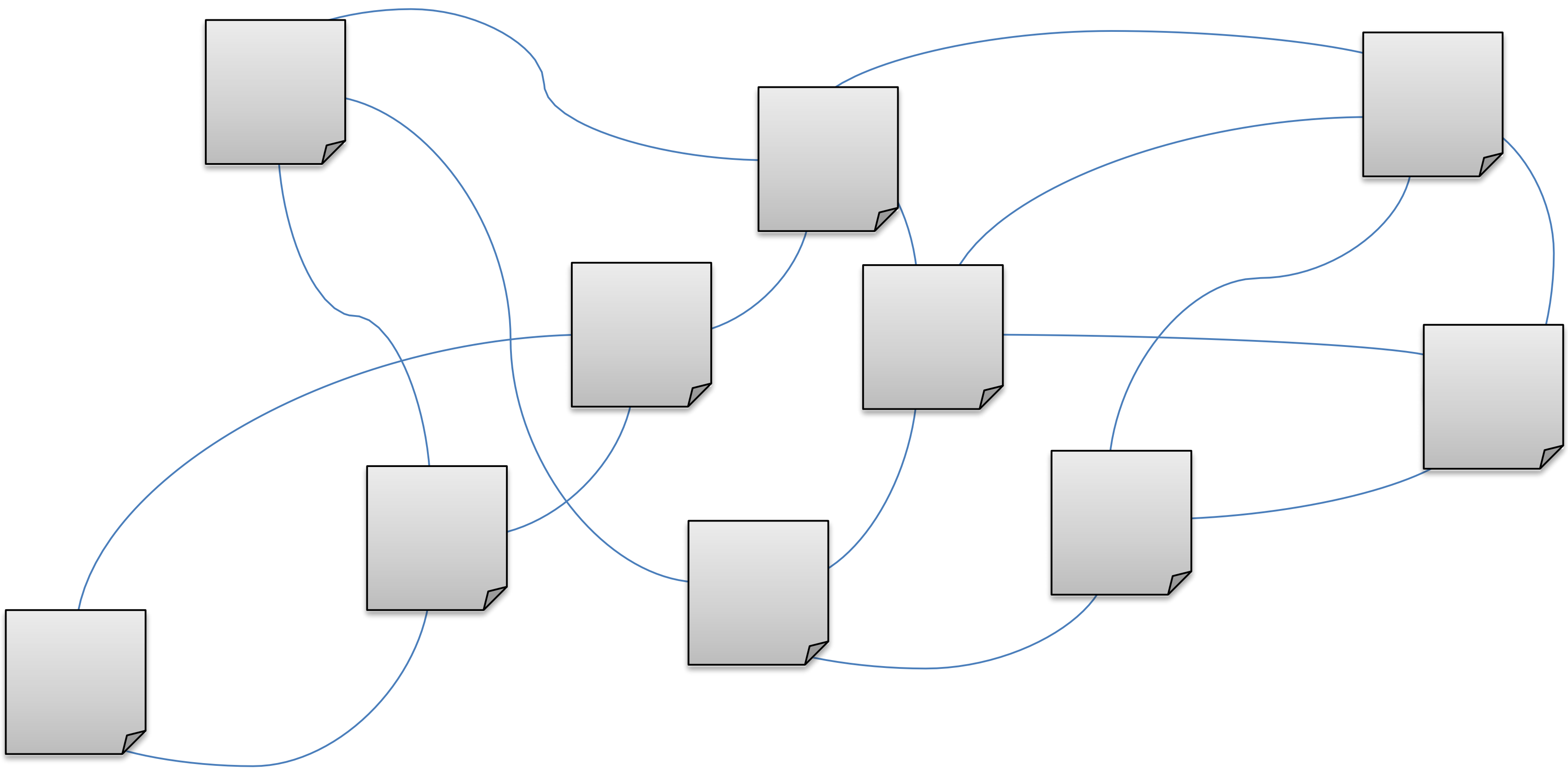
How can it be done?

Mask it again



How can it be done?

Allow the verifier to open two connecting masks



Properties of the Protocol

- The verifier learns nothing!
  - every time the verifier picks two masks to open, he already knows that they will be of different colour
- The prover cannot cheat
  - if there exists a connecting pair that cannot be coloured, there exists a probability (1/E, where E is the number of edges) that the verifier would choose to open that
  - for n rounds, the probability of successfully cheating is $(1 - 1/E)^n$

How to apply to the idea in Identification

- In the ElGamal Encryption Scheme, the public key is of the form, (Y, g, p) and the secret key is x such that ($Y = g^x \bmod p$)
- Example, $g = 10$, $p = 23$, $x = 3$.
- Then $Y = 11$
- Public Key (11, 10, 23)
- Secret Key (3)
- How to prove that you know the secret key?

Can I prove to you that I know the secret key?

- The prover picks a random number r
- Compute $T = g^r \bmod p$
- The prover sends T to the verifier
- At this stage, the verifier has
  - g, p, Y, T
- The verifier can ask one of the following two questions:
  - What is r?
  - What is r + x?

Zero-Knowledge Proof of x

- Verifier has (g, p, Y, T)
- If the verifier asks what is the value of r
  - The prover returns r
  - The verifier checks if $T = g^r \bmod p$
- If the verifier asks what is the value of r+x
  - The prover returns $z = r+x$
  - The verifier checks if $g^z = TY \bmod p$

What is the probability of cheating?

- If the prover want to answer the question “what is the value of r”
  - He/she can answer as long as T is computed correctly
- If the prover wants to answer the question “what is the value of r+x” without knowing x…
  - He/she can cheat by generating T as g^z/Y mod p for a randomly generated z…
  - In this case, he cannot answer the value of r…
- Cheating probability: 0.5

What is the probability of cheating?

- If someone can always (e.g., in 100 interactions) answer the question correctly, it is very likely that he/she knows x

Does it leak information?

- For every interaction, the verifier gets either r or r+x
- There are two unknowns (r and x) and one equation
- Thus, it does not leak any information

- The above protocol requires a lot of rounds to reduce the cheating probability to a negligible level
- Question: is there any protocol that can achieve the same goal in just 1 round

Schnorr ID Scheme

- Among the most well-known public-key ID schemes
- Based on the hardness of the Discrete Logarithm problem
- Similar schemes were developed later based on other hard computational problems
  - Guillou-Quisquater ID scheme based on RSA

Schnorr ID Scheme

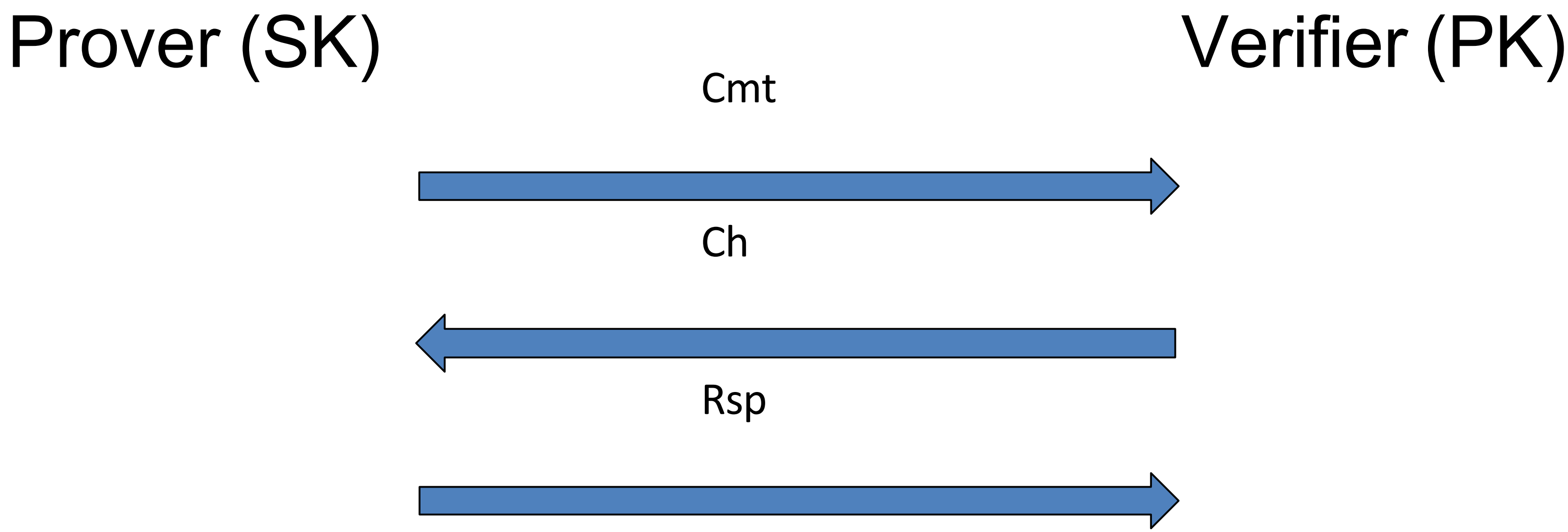
- General idea: the client (or prover) proves to the server (or verifier) that s/he has the SK corresponding to a PK
  - Completeness: with the correct SK, the client can always pass the verification
  - Soundness: the verification would fail if the prover does not have SK
  - Zero-knowledge: no information about SK is leaked in the identification process

Zero-knowledge

- How to define it?
- Use simulation: what the verifier can see in the proof can be simulated by itself
  - The distribution of a real proof is indistinguishable from that of a simulated proof

Schnorr ID Scheme

- An example of the so called Σ-protocol



Schnorr ID Scheme

- Let p and q be large prime numbers
  - $q \mid p-1$
- Let g be a generator (or primitive element) of a group G with order q
  - G is a subgroup of Z^*_p
- The parameter is (p, q, g)

Schnorr ID Scheme

(PK, SK) = (y, x) where $y = g^x \bmod p$

(1)Commitment: $a=g^r \bmod p$, where r is a random number from Z_q.

(2)Challenge: c, a random number from Z_q, selected by V.

(3)Response: $z=r+xc \bmod q$, computed by P using x.

Prover P (x)		**Verifier V (y)**
(1)	$a=g^r \bmod p$ $\longrightarrow$	
(2)	c $\longleftarrow$	
(3)	$z=r+xc \bmod q$ $\longrightarrow$	accept iff $g^z=ay^c \bmod p$

Schnorr ID Scheme

Prover P (x) **Verifier V (y)**

(1) $a = g^r \bmod p$ $\longrightarrow$

(2) c $\longleftarrow$

(3) $z = r + xc \bmod q$ $\longrightarrow$

accept iff $g^z = ay^c \bmod p$

- Completeness: if P and V are both honest, V will always accept since

  $g^z = g^{r+xc} = ag^{xc} = ay^c \bmod p$

Schnorr ID Scheme

<u>Prover P</u> (x)			**<u>Verifier V (y)</u>**
	(1)	$a = g^r \bmod p$ $\longrightarrow$	
	(2)	c $\longleftarrow$	
	(3)	$z = r + xc \bmod q$ $\longrightarrow$	
			accept iff $g^z = ay^c \bmod p$

- Soundness: if the verifier accepts, then z must have been computed using the correct secret key x

$g^z = ay^c \Rightarrow g^z = g^r y^c \Rightarrow g^z = g^r g^{xc} \Rightarrow z = r + xc \bmod q$

Schnorr ID Scheme

Prover P (x) | **Verifier V (y)**

(1) $a = g^r \bmod p$ →

(2) c ←

(3) $z = r + xc \bmod q$ →

accept iff $g^z = ay^c \bmod p$

- Zero-Knowledge:
  - Anyone can simulate a valid communication transcript (a, c, z) that satisfies $g^z=ay^c \bmod p$
  - Pick random c and z from Z_q, and compute $a = g^z/y^c \bmod p$
  - The simulated transcript has the same distribution as a normal transcript between P and V

Schnorr ID Scheme

Prover P (x) — **Verifier V (y)**

(1) $a = g^r \bmod p$ $\longrightarrow$

(2) c $\longleftarrow$

(3) $z = r + xc \bmod q$ $\longrightarrow$

accept iff $g^z = ay^c \bmod p$

- What is the chance that the prover can cheat in this protocol?

Non-interactive Zero-Knowledge

- Zero-Knowledge proofs can be made non-interactive
  - Known as NIZK proofs
- NIZK becomes popular in recent years
  - Zero Coin
  - Z-Cash
  - Based on advanced NIZK proof techniques (zk-SNARK)