Advanced Tools from Modern Cryptography

Lecture 9

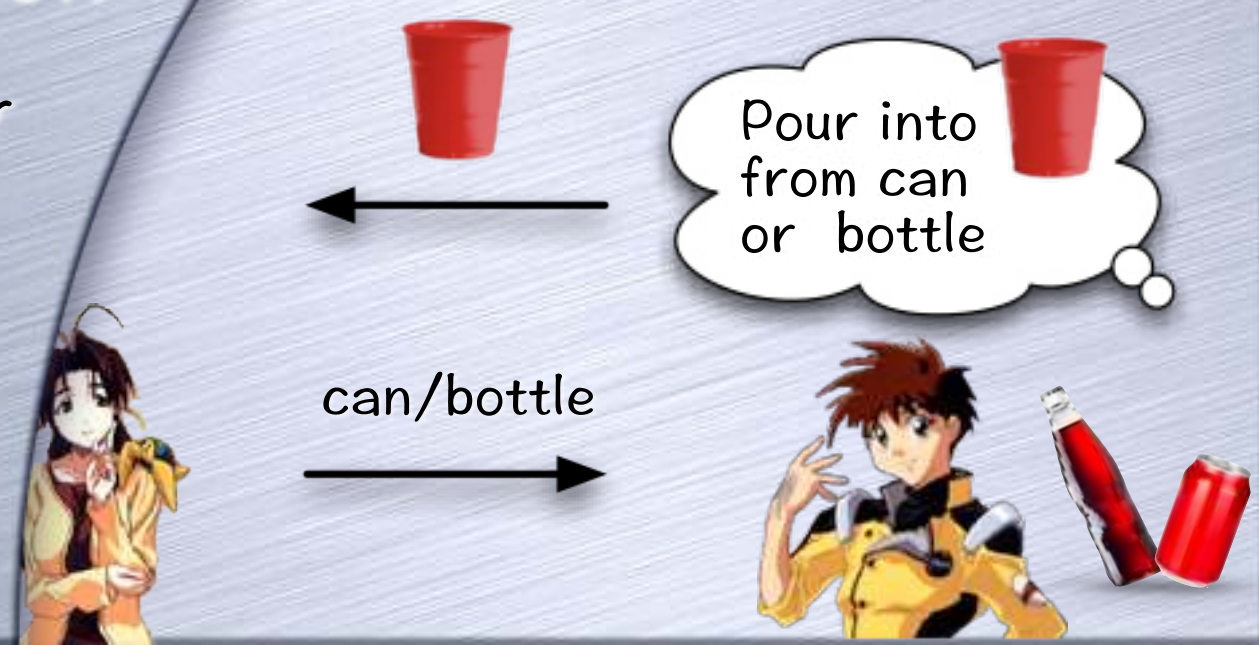
Zero-Knowledge Proofs

Zero-Knowledge Proof

- In cryptographic settings, often need to be able to verify various claims
  - e.g., 3 encryptions A,B,C are of values a,b,c s.t. a=b+c
  - Proof 1: Reveal a,b,c and how they get encrypted into A,B,C
  - Proof 2: Without revealing anything at all about a,b,c except the fact that a=b+c ?
    - Zero-Knowledge Proof!
- Important application to secure multi-party computation: to upgrade the security of MPC protocols from security against passive corruption to security against active corruption
  - (Next time)

Interactive Proofs
An Example

- Soft-drink in bottle or can
  - Prover claims: a soft-drink in bottle in a can tastes different from the same in a bottle
- An interactive proof:
  - prover tells whether the cup was filled from can or bottle
  - repeat till verifier is convinced



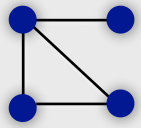
Interactive Proofs
An Example

- Graph Non-Isomorphism
  - Prover claims:
    G_0 **not** isomorphic to G_1
- An interactive proof:
  - prover tells whether
    G^* is an isomorphism
    of G_0 or G_1
  - repeat till verifier
    is convinced

Isomorphism: Same graph can be represented as a matrix in different ways:

e.g. $G_0 = \begin{matrix} 0&1&0&1 \\ 1&0&0&1 \\ 0&0&0&1 \\ 1&1&1&0 \end{matrix}$ and $G_1 = \begin{matrix} 0&1&0&1 \\ 1&0&1&1 \\ 0&1&0&0 \\ 1&1&0&0 \end{matrix}$

both are isomorphic to the graph represented by the drawing



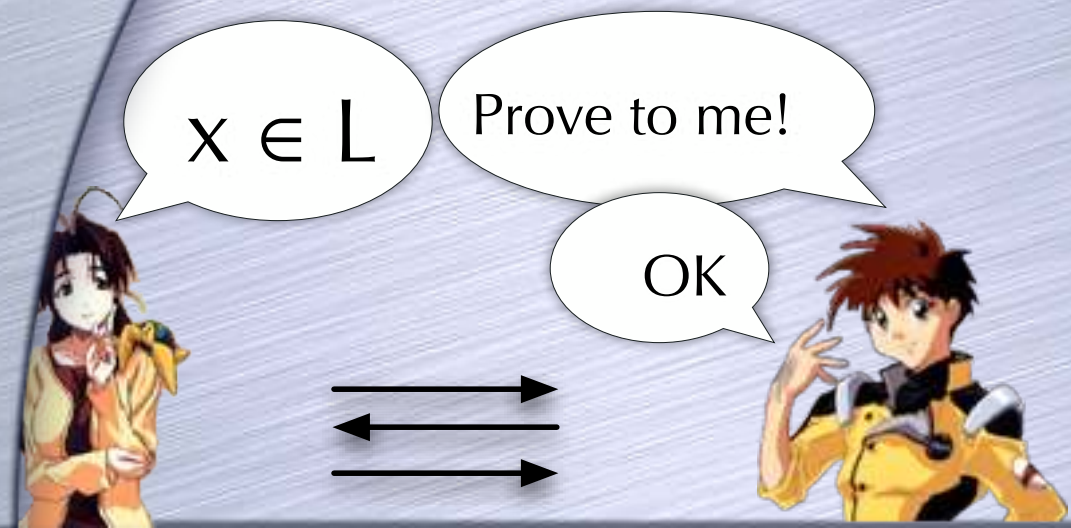
Set G^* to be $\pi(G_0)$ or $\pi(G_1)$ (π random)

G^*

G_0/G_1

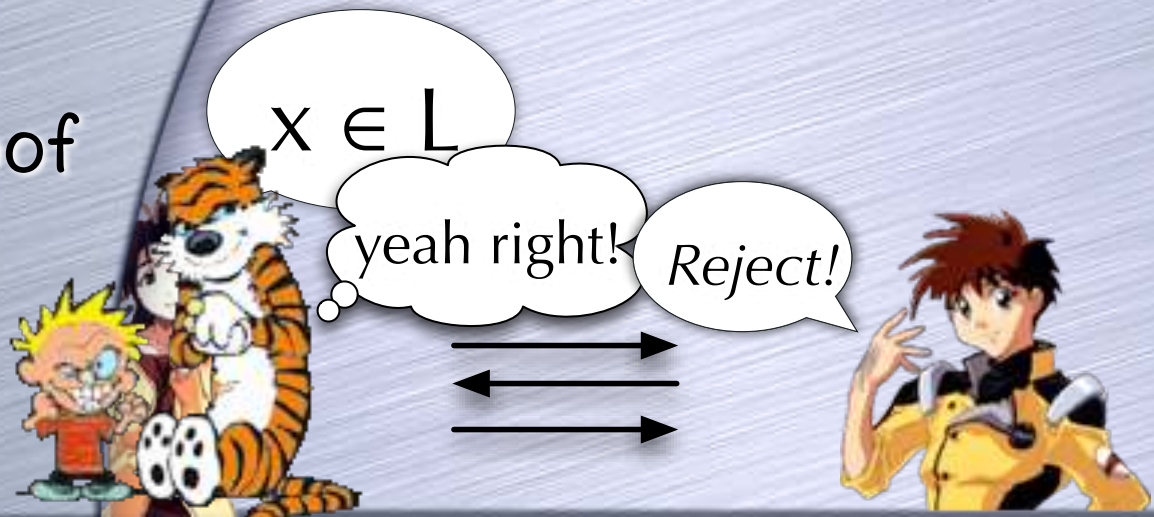
Interactive Proofs

- **Prover** wants to convince **verifier** that x has some property
  - i.e. x belongs to some set L ("language" L)
- All powerful prover (for now), and a computationally bounded verifier



Interactive Proofs

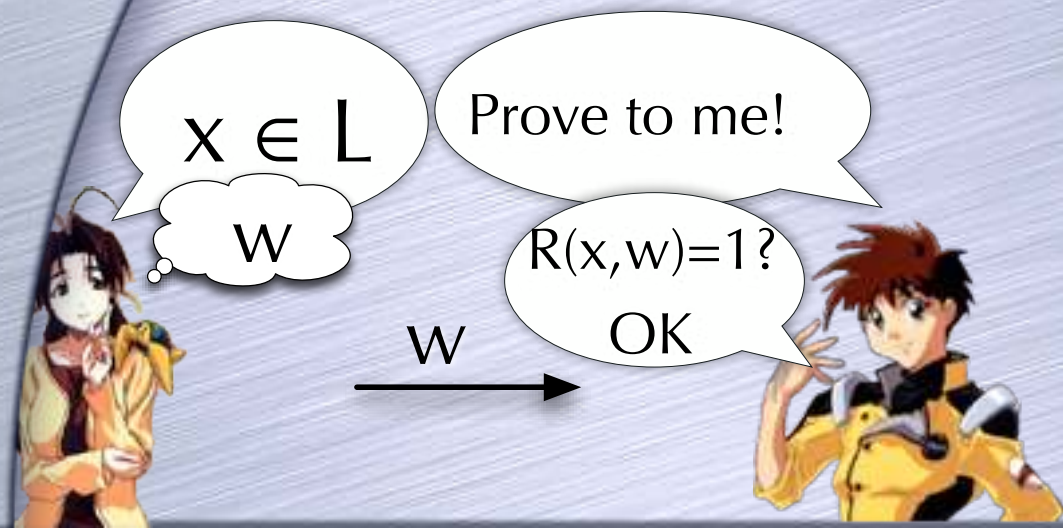
- Completeness
  - If x in L, honest Prover will convince honest Verifier
- Soundness
  - If x not in L, honest Verifier won't accept any purported proof



Proofs for an NP Language

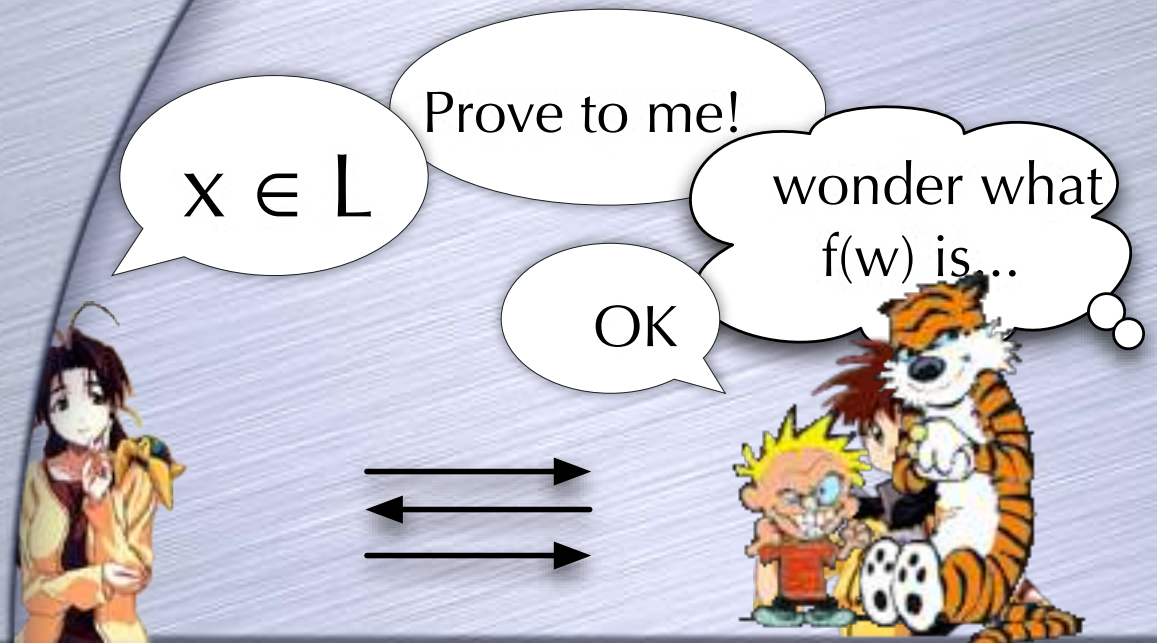
- NP language L
  - $x \in L$ iff $\exists w\ R(x,w)=1$ (for R in P)
    - e.g. Graph Isomorphism
- IP protocol
  - prover just sends w
- But what if prover doesn't want to reveal w?

NP is the class of languages which have non-interactive and deterministic proof-systems



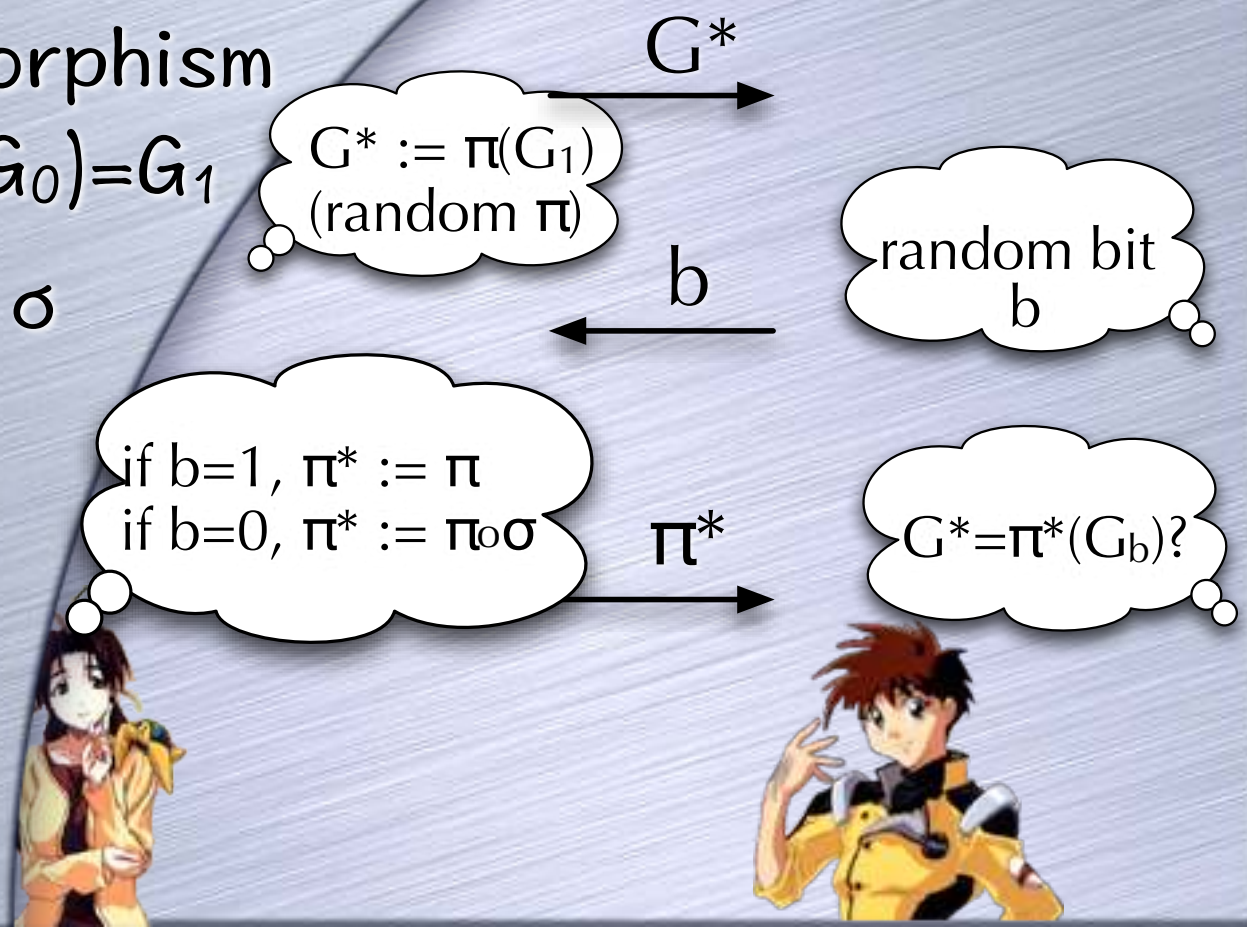
Zero-Knowledge Proofs

- Verifier should not gain **any** knowledge from the honest prover
  - except whether x is in L
- How to formalise this?
  - Simulation!



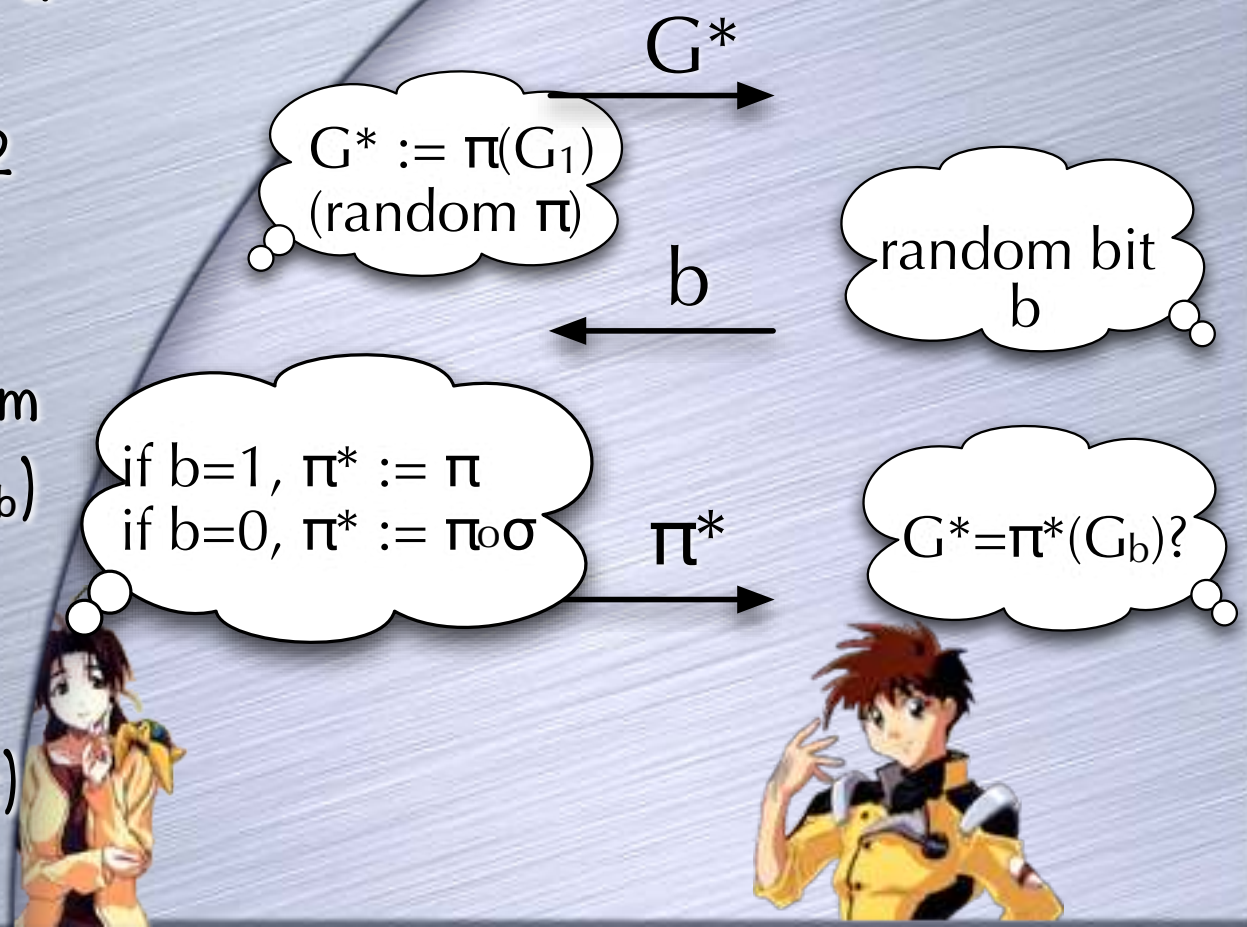
An Example

- Graph Isomorphism
  - (G_0, G_1) in L iff there exists an isomorphism σ such that $\sigma(G_0)=G_1$
- IP protocol: send σ
- ZK protocol?



An Example

- Why is this convincing?
  - If prover can answer both b's for the same G* then $G_0 \sim G_1$
  - Otherwise, testing on a random b will leave prover stuck w.p. 1/2
- Why ZK?
  - Verifier's view: random b and π^* s.t. $G^* = \pi^*(G_b)$
  - Which he could have generated by himself (whether $G_0 \sim G_1$ or not)



The Legend of William Tell

A Side Story

Bob: *William Tell is a great marksman!*

Charlie: *How do you know?*

Bob: *I just saw him shoot an apple placed on his son's head! See this!*

Charlie: *That apple convinced you? Anyone could have made it up!*

Bob: *But I saw him shoot it...*

The Legend of William Tell

A Side Story

Bob: *William Tell is a great marksman!*

Charlie: *How do you know?*

Bob: *I just saw him shoot an apple placed on his son's head! See this!*

Charlie: *That apple convinced you? Anyone could have made it up!*

Bob: *But I saw him shoot it...*

Bob: G_0 and G_1 are isomorphic!

Charlie: How do you know?

Bob: Alice just proved it to me! See this:

$$G^*, b, \pi^* \text{ s.t. } G^*=\pi^*(G_b)$$

Charlie: That convinced you? Anyone could have made it up!

Bob: But I picked b at random and she had no trouble answering me...

Simulation

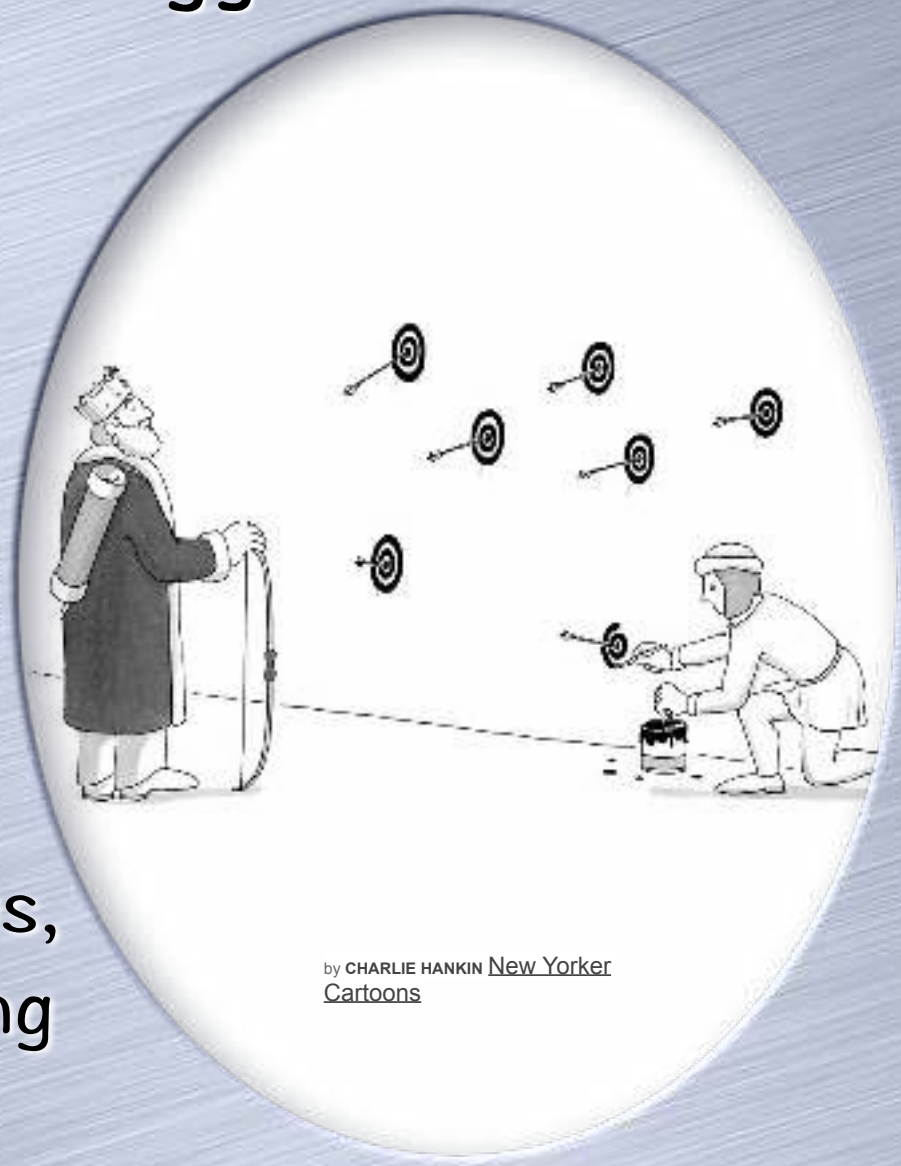
Another Analogy

- Shooting arrows at targets drawn randomly on a wall

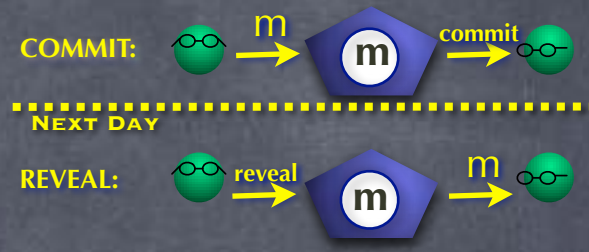
vs.

- Drawing targets around arrows shot randomly on to the wall

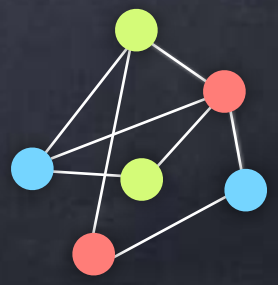
- Both produce identical views, but one of them is convincing of marksmanship



Commitment

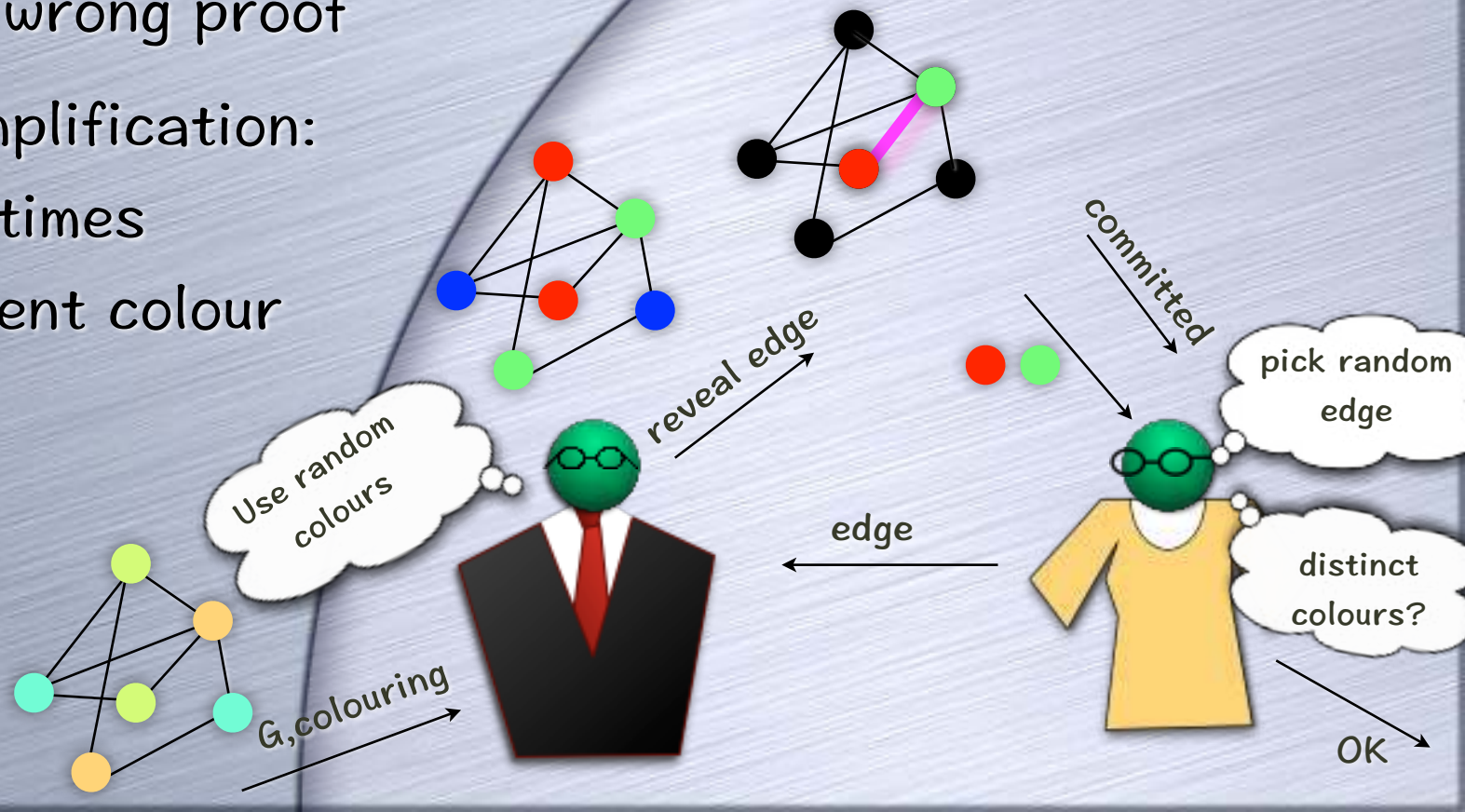


- Recall the functionality of **Commitment**:
  - Committing to a value: Alice puts the message in a box, locks it, and sends the locked box to Bob, who learns nothing about the message
  - Revealing a value: Alice sends the key to Bob. At this point she can't influence the message that Bob will get on opening the box.
- Example implementation in the Random Oracle Model: Commit(x) = H(x,r) where r is a long enough random string, and H is a random hash function (available as an oracle) with a long enough output. To reveal, send (x,r).
  - ⚠ ROM is a heuristic model: Can do provably impossible tasks in this model!
- An Example: To prove that the nodes of a graph can be coloured with at most 3 colours, so that adjacent nodes have different colours



A ZK Proof for Graph Colourability

- Uses commitment functionality
- At least 1/#edges probability of catching a wrong proof
- Soundness amplification: Repeat many times with independent colour permutations

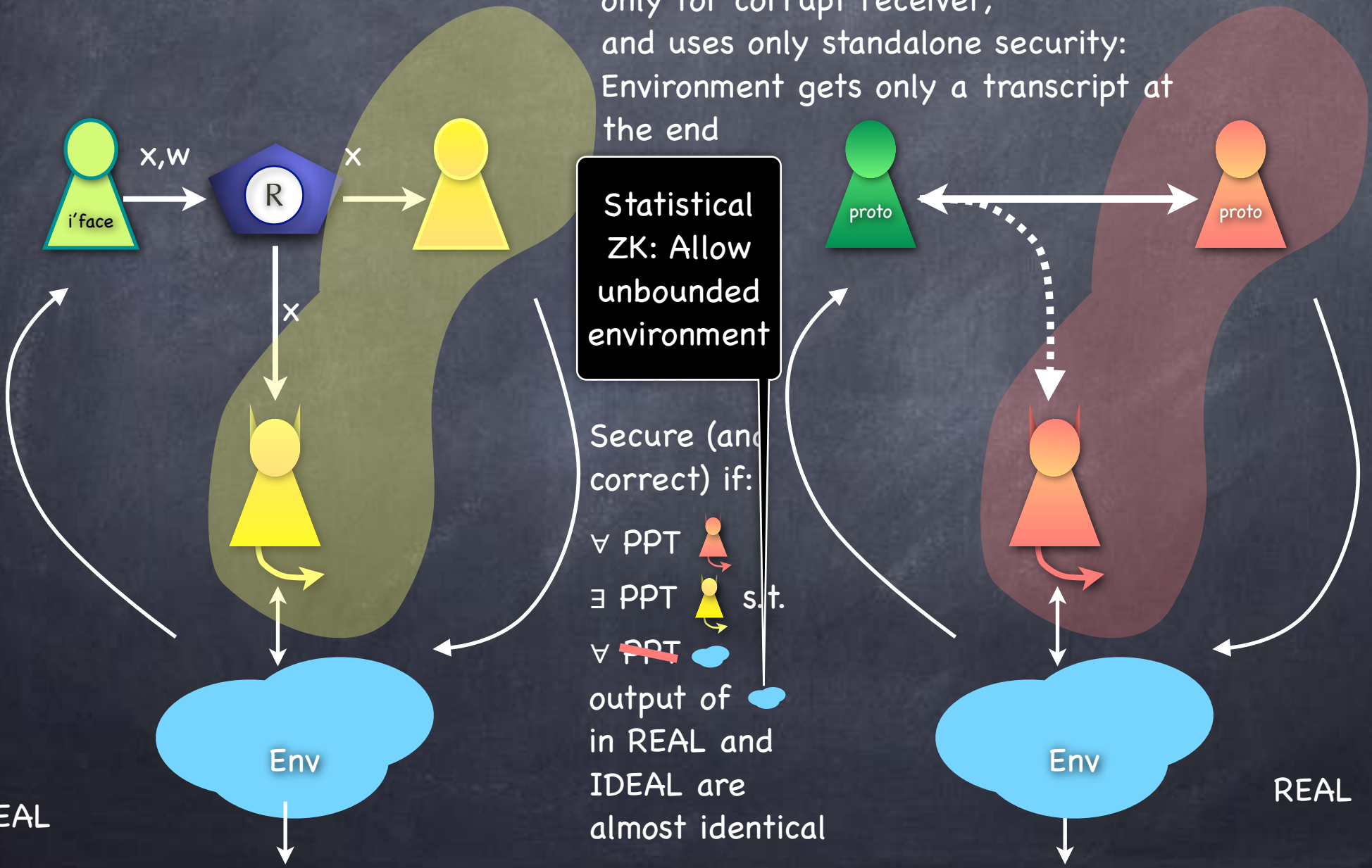


ZK Proofs Vocabulary

- Statements: Of the form "$\exists w$ s.t. relation R(x,w) holds", where R defines a class of statements, and x specifies the particular statement (which is a common input to prover and verifier)
  - e.g., Given a graph G, $\exists$ a colouring ϕ s.t. Valid(G,ϕ) holds
  - The relation R can be efficiently verified (polynomial time in size of x)
    - Set L = { x | $\exists w$ R(x,w) holds } is a language in NP
  - w is called a "witness" for $x \in L$
- **Completeness**: If prover & verifier are honest, for all $x \in L$, and prover given a valid witness w, verifier will always accept
- **Soundness**: If $x \notin L$, no matter what a cheating prover does, an honest verifier will reject (except with negligible probability)
  - Proof-of-Knowledge: A stronger soundness notion
- **Zero-Knowledge**: A (corrupt) verifier's view can be simulated (honest prover, $x \in L$)
- Soundness can be required to hold even against computationally unbounded provers
  - ZK Argument system: Like a ZK proof system, but soundness only against PPT adversaries

ZK Property

Classical definition uses simulation only for corrupt receiver;
and uses only standalone security:
Environment gets only a transcript at the end



Statistical ZK: Allow unbounded environment

Secure (and correct) if:

∀ PPT

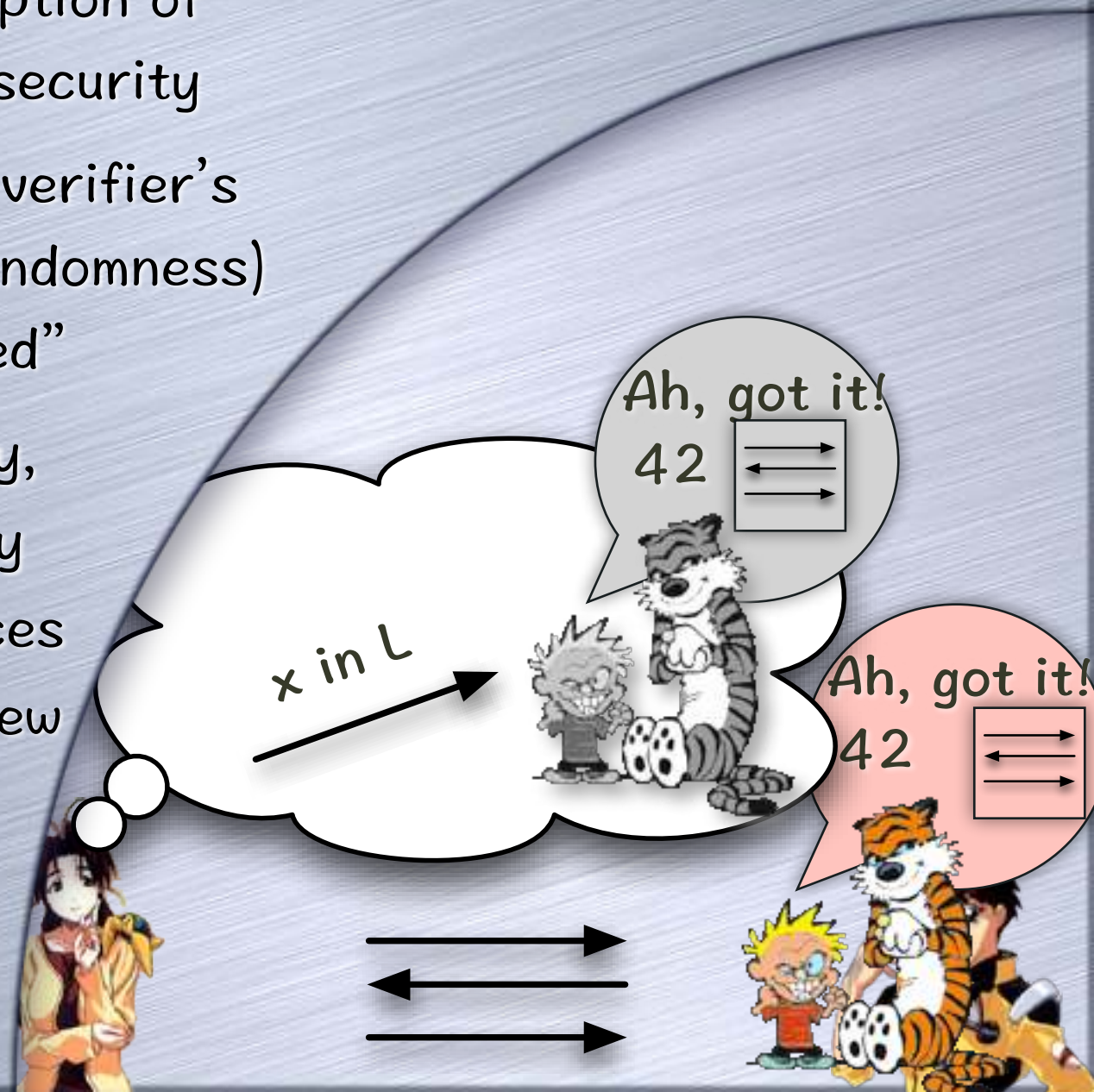
∃ PPT s.t.

∀ ~~PPT~~

output of in REAL and IDEAL are almost identical

In Other Pictures…

- Simulation only for corruption of verifier and stand-alone security
- ZK Property: A corrupt verifier's view (i.e., transcript + randomness) could have been "simulated"
  - ∀ adversarial strategy, ∃ a simulation strategy which, ∀x ∈ L, produces an indistinguishable view
- Completeness and soundness defined separately



Two-Sided Simulation

- Require simulation also when prover is corrupt
  - Then simulator is a witness extractor
- Adding this (in standalone setting) makes it an Argument of Knowledge