

# Zero Knowledge



# What is a Zero-Knowledge Proof?

A convincing demonstration that some statement is true—  
*without revealing anything* beyond the fact that it's true!

**Example:** Suppose you knew that these two pens are different.  
How could you convince me they're different without telling me  
what is different about them?

# Zero-Knowledge Proof for “Where’s Waldo?” (or “where’s the puffin among the penguins?”)

How could you convince me that there is a puffin in this picture without revealing its location (or anything else)?



Computer Scientist Explains One Concept in 5 Levels of Difficulty | WIRED

## Goldwasser and Micali win Turing Award

Team honored for 'revolutionizing the science of cryptography.'

Abby Abazorius, CSAIL  
March 13, 2013

Including introducing zero-knowledge proofs



Chris Peikert's PhD advisor



Shafi Goldwasser

Silvio Micali

# How are Zero-Knowledge Proofs Useful?

- **Electronic money / Blockchain:** Prove that your account has enough money for a transaction, without revealing your balance.
- **Group signatures:** Prove that a member of a certain group signed a message, without revealing which one did.
  - E.g., give students keycard access to restricted areas without tracking individual students.
- **Multi-party computation:** Compute aggregate functions of private data (e.g., medical records), without revealing individual data.

# The Model

Both parties  
know  $x$  and  $L$



Let's put aside zero knowledge for a moment and recall  
**verifying a certificate for an instance  $x$  of a language  $L$  in NP.**



Given an instance  $x$   
(of an NP-language  $L$ ),  
I send a certificate  $c$

$c$



I run a verification  
algorithm  $V(x, c)$



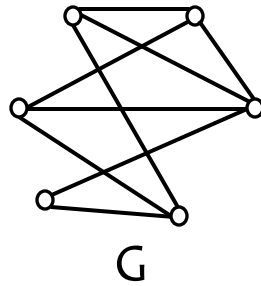
Recall that a language  $L$  is in NP if:

- $x \in L \Rightarrow$  **Merlin** can give **Arthur** a convincing “proof” that  $x \in L$ ,  
i.e., exists certificate  $c$  so that  $V(x, c)$  accepts
- $x \notin L \Rightarrow$  **Merlin** can't “fool” **Arthur** into accepting that  $x \in L$ ,  
i.e., for any  $c$ ,  $V(x, c)$  rejects.

**Merlin**  
“the prover”  
All-powerful but  
untrustworthy

**Arthur**  
“the verifier”  
Restricted to  
poly-time  
computation

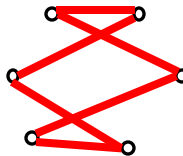
# Hamiltonian Cycle



Let's put aside zero knowledge for a moment and recall **verifying a certificate for an instance  $x$  of a language  $L$  in NP**



I send a  
Hamiltonian  
Cycle in  $G$



I check it, and am  
convinced that  $G$   
has a Hamiltonian  
Cycle



This is **NOT** a ZK proof because Arthur learns an actual Ham Cycle (rather than just the fact that  $G$  has a Ham Cycle)!

Merlin  
“the prover”  
All-powerful but  
untrustworthy

**Can we make this into a ZK proof?**

**YES!**

By using randomness and interaction.

Arthur  
“the verifier”  
Restricted to  
poly-time  
computation

\*Actually it's possible to make it non-interactive (we won't show)



# Requirements for a protocol to be a ZK proof

- **“Completeness”**: If  $x \in L$ , **Merlin** can cause **Arthur** to accept.
- **“Soundness”**: If  $x \notin L$ , then **Merlin** cannot “fool” **Arthur** into accepting, except with some *small probability*. (← new, needed weakening)
- **“Zero knowledge”**: **Arthur** “learns nothing” but the fact that  $x \in L$ .

**Q:** *But what exactly does that mean? How to define it?*

**A:** Given that  $x \in L$ , whatever **Arthur** saw from **Merlin** could have been generated by **Arthur** on his own, without ever interacting with **Merlin**.

↳ If **Merlin** uses randomness, **Arthur** could have generated messages from the same probability distribution as **Merlin's** messages.

- **“Efficiency”**: **Arthur** runs in polynomial time (in the size of  $x$ ).



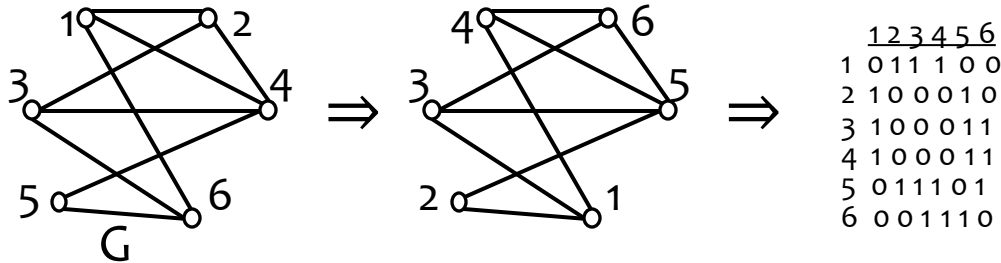
# A Zero-Knowledge Proof for HamCycle [Blum '86]



Some academic descendants of Blum in our department:  
Chris Peikert, Euiwoong Lee, Ben Fish, Wei Hu

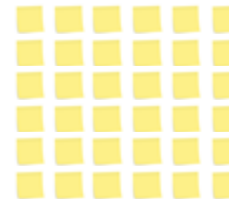
# A Zero-Knowledge Proof for HamCycle [Blum '86]

Merlin convincing Arthur that G has a Ham Cycle without revealing anything else



I am supposed to:

- Randomly shuffle vertices.
- Write adjacency matrix.
- Cover each entry with a post-it note, and send!



I randomly choose one *challenge*:

- **“reveal all”** or
- **“reveal cycle”**



**“reveal all”**

If **“reveal all”**, I reveal the *entire* adjacency matrix *and* the shuffling permutation

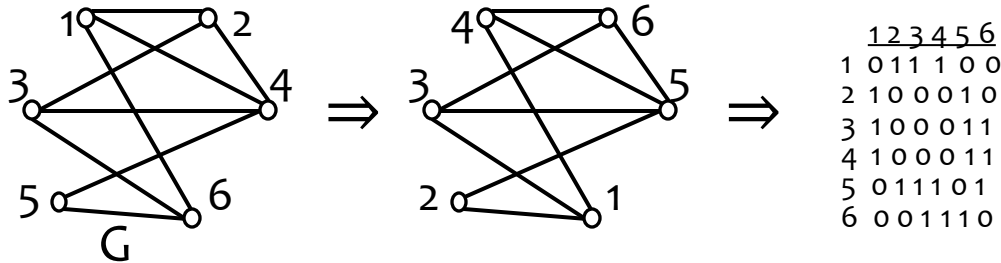
	1	2	3	4	5	6
1	0	1	1	1	0	0
2	1	0	0	0	1	0
3	1	0	0	0	1	1
4	1	0	0	0	1	1
5	0	1	1	1	0	1
6	0	0	1	1	1	0

permutation: 4, 6, 3, 5, 2, 1

Accept if this matches original graph G

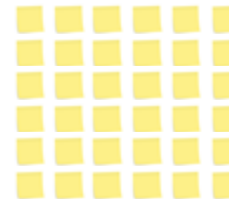
# A Zero-Knowledge Proof for HamCycle [Blum '86]

Merlin convincing Arthur that  $G$  has a Ham Cycle without revealing anything else



I am supposed to:

- Randomly shuffle vertices.
- Write adjacency matrix.
- Cover each entry with a post-it note, and send!



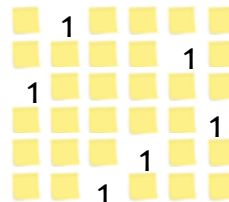
I randomly choose one *challenge*:

- **“reveal all”** or
- **“reveal cycle”**



If **“reveal cycle”**,  
I reveal only the  
(shuffled) Ham  
Cycle

**“reveal cycle”**



Accept if revealed  
entries are 1s, and  
correspond to a  
Ham Cycle  
(ignoring  $G$  itself)

# A Zero-Knowledge Proof for HamCycle [Blum '86]

Merlin convincing Arthur that G has a Ham Cycle without revealing anything else


- **“Completeness”**: If G has a Ham Cycle, Merlin can cause Arthur to accept: *Merlin correctly follows protocol*
- **“Soundness”**: If G has **no** Ham Cycle, then Merlin “fools” Arthur into accepting with probability  $\leq \frac{1}{2}$ : *If no Ham Cycle then either Merlin's adj matrix*
  - ① *Isn't a shuffling of G*  $\leftarrow$  *Merlin fails "reveal all"*
  - OR
  - ② *Doesn't have Ham Cycle*  $\leftarrow$  *Merlin fails "reveal cycle"*


How can we reduce this “fooling” probability to near 0? *Repeat protocol!*

# A Zero-Knowledge Proof for HamCycle [Blum '86]

Merlin convincing Arthur that  $G$  has a Ham Cycle without revealing anything else

- **“Zero knowledge”**: If  $G$  has a Ham Cycle, then **Arthur**, **without ever interacting with Merlin**, could have generated messages from the same probability distribution as Merlin's.

“reveal all”: random perm + shuffling of  $G$  

“reveal cycle”: cycle edges (4s) w/ randomly permuted vertices 

- **“Efficiency”**: **Arthur** runs in polynomial time 

# Digital Commitment Schemes

A digital version of this ZK proof requires a “**commitment scheme**”.

I.e., **Merlin** “commits” to bits without revealing them to **Arthur**, and can later reveal any of them as needed—without being able to *change* them.

**This is possible!**  
(under cryptographic assumptions)

We won't prove it, but here's some intuition:

- **Merlin** “commits” to primes  $p, q$  by sending **Arthur** their product  $n=pq$ .
- **Arthur** can't factor efficiently, so doesn't know  $p$  and  $q$ .
- **Merlin** can later reveal  $p, q$ . **Arthur** checks that they are prime and that their product is  $n$ . No other numbers satisfy these properties, so Merlin could not have changed them!

(Needs to be extended to allow **Merlin** to commit to a desired *bit*.)

# This implies a ZK proof for ANY problem in NP!

(under cryptographic assumptions)

## How?

Let  $L$  be a language in NP.

We want a ZK proof that  $x \in L$ .

Let  $f$  be a p.t.m.reduction from  $L$  to HC  
(exists since HC is NP-complete).

So,  $x \in L \Leftrightarrow f(x) \in \text{HC}$ .

To ZK-prove that  $x \in L$ , use the ZK proof for HC to prove that  $f(x) \in \text{HC}$ .

First ZK proof for an NP-C problem (with larger error) [Goldreich-Micali-Wigderson '86]

This includes NP-complete problems, problems in P, decision versions of discrete log and factoring, ...



Won the Turing Award last week!

For “reshaping our understanding of the role of randomness in computation”





# Beyond NP

**Theorem** (we won't show): Under cryptographic assumptions, the set of problems that have a **ZK proof** is *equivalent* to the set of problems in **PSPACE**, i.e., solvable w/ **polynomial space**. [Lund-Karloff-Fortnow-Nisan-Shamir'92]

It is known that  $\mathbf{NP} \subseteq \mathbf{PSPACE}$ , and it is conjectured that  $\mathbf{NP} \neq \mathbf{PSPACE}$ .

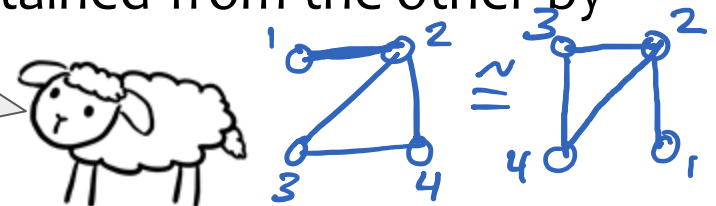
There are also problems that seem to be outside of NP that have ZK proofs, *unconditionally*!

**Example:** There is a ZK proof for **Graph Non-Isomorphism**:

Given two graphs, are they “different” (non-isomorphic)? [GMW '86]

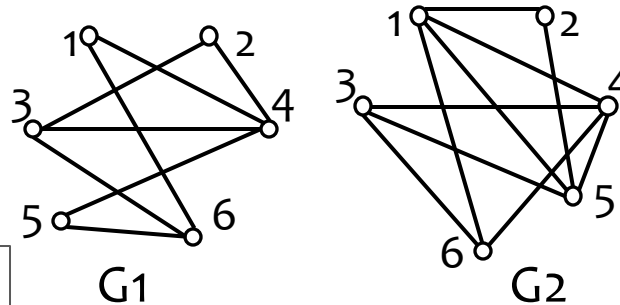
Two graphs are *isomorphic* if one can be obtained from the other by permuting the vertices.

This is reminiscent  
of the pen example



# A ZK Proof for Graph Non-Isomorphism

Merlin can convince Arthur that  $G_1$  and  $G_2$  are non-isomorphic without revealing anything else.



I say whether  
it's a  
scrambling of  
 $G_1$  or  $G_2$

	1	2	3	4	5	6
1	0	1	1	1	0	0
2	1	0	0	0	1	0
3	1	0	0	0	1	1
4	1	0	0	0	1	1
5	0	1	1	1	0	1
6	0	0	1	1	1	0

(a scrambling  
of  $G_1$ , generated  
by Arthur)

I randomly choose  
 $G_1$  or  $G_2$ , randomly  
scramble the  
vertices, and send  
the resulting  
adjacency matrix!

It's  $G_1$ !



This is reminiscent  
of the pen example



I accept if  
Merlin got the  
right answer

# A ZK Proof for Graph Non-Isomorphism

Merlin can convince Arthur that  $G_1$  and  $G_2$  are non-isomorphic without revealing anything else.

- **“Completeness”**: If  $G_1$  and  $G_2$  are non-isomorphic, Merlin can cause Arthur to accept: *If non-iso, Merlin can tell which graph was scrambled, so gets right answer.*
- **“Soundness”**: If  $G_1$  and  $G_2$  are isomorphic, then Merlin “fools” Arthur into accepting with probability  $\leq \frac{1}{2}$ : *Merlin can't tell which graph Arthur scrambled. Arthur picked randomly so  $\frac{1}{2}$  chance of guessy right*

How can we reduce this “fooling” probability to near 0? *Repeat protocol*

# A ZK Proof for Graph Non-Isomorphism

Merlin can convince Arthur that  $G_1$  and  $G_2$  are non-isomorphic without revealing anything else.

- **“Zero knowledge”**: If  $G_1$  and  $G_2$  are non-isomorphic, then whatever **Arthur** saw from **Merlin** could have been **generated by Arthur on his own, without ever interacting with Merlin**.

*Arthur already knew if  $G_1$  or  $G_2$ .*

- **“Efficiency”**: **Arthur** runs in polynomial time ✓

# Admin

Exam Review Session led by Daphne:  
Wednesday April 24th, 7-9pm, BBB 1670.

HW11 is due Tuesday 4/23 at 8pm.

Reminder: Filling out the course evaluations is 1% of your grade, which is otherwise covered by the final exam. Remember to submit receipt on Gradescope.

# Next Monday: Last Lecture

Nicole Wein: **Online Algorithms**

Thatchaphol Saranurak: **Vertex Connectivity**

Chris Peikert: **Quantum-Secure Cryptography with Lattices**

Mark Brehob: **Caching**

See piazza post for more detail

