ZK Proofs (cntd.)
Composition
ZK Proofs (cntd.)

Composition

Lecture 16
An Example
An Example

Graph Isomorphism
An Example

Graph Isomorphism

$(G_0, G_1)$ in $L$ iff there exists an isomorphism $\sigma$ such that $\sigma(G_0) = G_1$
An Example

Graph Isomorphism

\((G_0, G_1) \text{ in } L \iff \text{there exists an isomorphism } \sigma \text{ such that } \sigma(G_0) = G_1\)

IP protocol: send \(\sigma\)
An Example

Graph Isomorphism

\((G_0, G_1)\) in \(L\) iff there exists an isomorphism \(\sigma\) such that \(\sigma(G_0) = G_1\)

IP protocol: send \(\sigma\)

ZK protocol
An Example

- **Graph Isomorphism**
  
  \((G_0, G_1)\) in L iff there exists an isomorphism \(\sigma\) such that \(\sigma(G_0) = G_1\)

- **IP protocol**: send \(\sigma\)

- **ZK protocol**
  
  Bob sees only \(b, \pi^*\) and \(G^*\) s.t. \(\pi^*(G_b) = G^*\)
An Example

**Graph Isomorphism**

\((G_0, G_1) in L\) iff there exists an isomorphism \(\sigma\) such that \(\sigma(G_0) = G_1\)

**IP protocol:** send \(\sigma\)  

**ZK protocol**

Bob sees only \(b, \pi^*\) and \(G^*\) s.t. \(\pi^*(G_b) = G^*\)
An Example

Graph Isomorphism

\((G_0, G_1)\) in L iff there exists an isomorphism \(\sigma\) such that \(\sigma(G_0) = G_1\)

IP protocol: send \(\sigma\)

ZK protocol

Bob sees only \(b, \pi^*\) and \(G^*\) s.t. \(\pi^*(G_b) = G^*\)

\[G^* := \pi(G_1)\] (random \(\pi\))
An Example

Graph Isomorphism

\( (G_0, G_1) \) in L iff there exists an isomorphism \( \sigma \) such that \( \sigma(G_0) = G_1 \)

IP protocol: send \( \sigma \)

ZK protocol

Bob sees only \( b, \pi^* \) and \( G^* \) s.t.

\[ \pi^*(G_b) = G^* \]
An Example

Graph Isomorphism

(G₀,G₁) in L iff there exists an isomorphism σ such that σ(G₀)=G₁

IP protocol: send σ

ZK protocol

Bob sees only b, π* and G* s.t. π*(G_b) = G*

G* := π(G₁) (random π)

random bit b
**An Example**

- **Graph Isomorphism**
  
  \((G_0, G_1)\) in \(L\) iff there exists an isomorphism \(\sigma\) such that \(\sigma(G_0) = G_1\)

- **IP protocol**: send \(\sigma\)

- **ZK protocol**
  
  Bob sees only \(b, \pi^*\) and \(G^*\) s.t. \(\pi^*(G_b) = G^*\)

\(G^* := \pi(G_1)\) (random \(\pi\))

\(\pi^* := \pi\) if \(b = 1\)

\(\pi^* := \pi \circ \sigma\) if \(b = 0\)
An Example

**Graph Isomorphism**

(G₀, G₁) in L iff there exists an isomorphism σ such that σ(G₀) = G₁

**IP protocol:** send σ

**ZK protocol**

Bob sees only b, π* and G* s.t. π*(G_b) = G*

**Random Bit Protocol**

- If b = 1, π* := π
- If b = 0, π* := π o σ

**Diagram:**

- G* := π(G₁)
- G* := π(G₁) (random π)
- b
- random bit
- π*
- G*

RECALL
An Example

Graph Isomorphism

\((G_0, G_1)\) in \(L\) iff there exists an isomorphism \(\sigma\) such that \(\sigma(G_0) = G_1\)

IP protocol: send \(\sigma\)

ZK protocol

Bob sees only \(b, \pi^*\) and \(G^*\) s.t. \(\pi^*(G_b) = G^*\)
The Legend of William Tell

A Side Story
The Legend of William Tell

A Side Story

Bob: William Tell is a great marksman!
The Legend of William Tell

A Side Story

Bob: William Tell is a great marksman!

Charlie: How do you know?
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!
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!
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!
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: 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: 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: 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:
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:

$$G^*, b, \pi^* \text{ s.t. } G^* = \pi^*(G_b)$$
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: 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...
Zero-Knowledge Proofs
Zero-Knowledge Proofs

Interactive Proof
Zero-Knowledge Proofs

- Interactive Proof
- Complete and Sound
Zero-Knowledge Proofs

- Interactive Proof
- Complete and Sound

ZK Property:
Zero-Knowledge Proofs

- Interactive Proof
- Complete and Sound

ZK Property:
Zero-Knowledge Proofs

Interactive Proof

Complete and Sound

ZK Property:
Zero-Knowledge Proofs

Interactive Proof
- Complete and Sound

ZK Property:
Zero-Knowledge Proofs

- Interactive Proof
- Complete and Sound

ZK Property:
Zero-Knowledge Proofs

Interactive Proof

- Complete and Sound

ZK Property:

- Verifier’s view could have been “simulated”

Ah, got it!
42
Zero-Knowledge Proofs

Interactive Proof
- Complete and Sound

ZK Property:
- Verifier’s view could have been “simulated”
Zero-Knowledge Proofs

Interactive Proof
- Complete and Sound

ZK Property:
- Verifier’s view could have been “simulated”
Zero-Knowledge Proofs

Interactive Proof

Complete and Sound

ZK Property:

Verifier’s view could have been “simulated”

x in L

Ah, got it!
42
Zero-Knowledge Proofs

Interactive Proof
- Complete and Sound

ZK Property:
- Verifier’s view could have been “simulated”
Zero-Knowledge Proofs

Interactive Proof

Complete and Sound

ZK Property:

Verifier’s view could have been “simulated”

For every adversarial strategy, there exists a simulation strategy.
ZK Property (in other pict's)

Secure (and correct) if:

\[ \forall \exists \text{s.t.} \forall \text{output of is distributed identically in REAL and IDEAL} \]
ZK Property (in other pict's)

Secure (and correct) if:

∀ output of is distributed identically in REAL and IDEAL
ZK Property (in other pict's)

Secure (and correct) if:
\[ \forall \exists \text{s.t. } \forall \quad \text{output of is distributed identically in REAL and IDEAL} \]
ZK Property (in other pict’s)

Classical definition uses simulation only for corrupt receiver;

Secure (and correct) if:

∀ output of is distributed identically in REAL and IDEAL
ZK Property (in other pict’s)

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

Secure (and correct) if:
\[ \forall \exists \text{ s.t. } \forall \text{ output of is distributed identically in REAL and IDEAL} \]
Secure (and correct) if:

\[ \forall s.t. \forall \exists \] output of is distributed identically in REAL and IDEAL.
SIM ZK

- SIM-ZK would require simulation also when prover is corrupt

Secure (and correct) if:

\[ \forall \exists \text{s.t.} \forall \text{output of is distributed identically in REAL and IDEAL} \]
SIM ZK

- SIM-ZK would require simulation also when prover is corrupt
- Then simulator is a witness extractor

Secure (and correct) if:

\[ \forall \exists \text{ s.t. } \forall \text{ output of is distributed identically in REAL and IDEAL} \]
SIM ZK

- SIM-ZK would require simulation also when prover is corrupt
- Then simulator is a witness extractor
- Adding this (in standalone setting) makes it a **Proof of Knowledge**

- Secure (and correct) if:
  \[
  \forall \exists \text{s.t.} \forall \text{output of is distributed identically in REAL and IDEAL}
  \]
A ZK Proof for Graph Colorability
A ZK Proof for Graph Colorability
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine

Uses random colors

G, coloring

pick random edge

committed

edge
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine
A ZK Proof for Graph Colorability

Uses a commitment protocol as a subroutine

Pick random edge
distinct colors?
Use random colors
G, coloring
revealed edge
committed

OK
A ZK Proof for Graph Colorability

- Uses a commitment protocol as a subroutine
- At least $1/m$ probability of catching a wrong proof

Use random colors

G, coloring

reveal edge
distinct colors?

pick random edge

committed

OK
A ZK Proof for Graph Colorability

- Uses a commitment protocol as a subroutine
- At least $1/m$ probability of catching a wrong proof
- Soundness amplification: Repeat say $mk$ times (with independent color permutations)
A Commitment Protocol
A Commitment Protocol

Using a OWP f and a hardcore predicate for it B
Using a OWP \( f \) and a hardcore predicate for it \( B \).

Satisfies only classical (IND) security, in terms of hiding and binding.
Using a OWP \( f \) and a hardcore predicate for it \( B \)
Satisfies only classical (IND) security, in terms of hiding and binding
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$

Satisfies only classical (IND) security, in terms of hiding and binding
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$
Satisfies only classical (IND) security, in terms of hiding and binding

$\text{random}_x$, $f(x) \oplus B(x)$
A Commitment Protocol

- Using a OWP $f$ and a hardcore predicate for it $B$
- Satisfies only classical (IND) security, in terms of hiding and binding

$\text{random } x$

$f(x), b \oplus B(x)$

committed
A Commitment Protocol

Using a OWP \( f \) and a hardcore predicate for it \( B \)

Satisfies only classical (IND) security, in terms of hiding and binding
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$

Satisfies only classical (IND) security, in terms of hiding and binding
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$

Satisfies only classical (IND) security, in terms of hiding and binding
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$

Satisfies only classical (IND) security, in terms of hiding and binding

$$f(x), b \oplus B(x)$$

committed

consistent?

random $x$

reveal $b$
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$

Satisfies only classical (IND) security, in terms of hiding and binding
A Commitment Protocol

Using a OWP $f$ and a hardcore predicate for it $B$

Satisfies only classical (IND) security, in terms of hiding and binding

Perfectly binding because $f$ is a permutation

\[ f(x), b \oplus B(x) \]

\[ \text{random } x \]

\[ \text{committed } x, b \]

\[ \text{consistent?} \]

\[ b \]

\[ \text{reveal} \]
Using a OWP $f$ and a hardcore predicate for it $B$.
Satisfies only classical (IND) security, in terms of hiding and binding.
Perfectly binding because $f$ is a permutation.
Hiding because $B(x)$ is pseudorandom given $f(x)$. 

A Commitment Protocol

$f(x), b \oplus B(x) \quad \quad \quad x, b \quad \quad \quad b \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad 

\textit{random} $x$
ZK Results
ZK Results

IP and ZK defined [GMR’85]
ZK Results

- IP and ZK defined [GMR’85]
- ZK for all NP languages [GMW’86]
ZK Results

- IP and ZK defined [GMR’85]
- ZK for all NP languages [GMW’86]
- Assuming one-way functions exist
ZK Results

- IP and ZK defined [GMR’85]
- ZK for all NP languages [GMW’86]
  - Assuming one-way functions exist
- ZK for all of IP [BGGHKMR’88]
ZK Results

- IP and ZK defined [GMR’85]
- ZK for all NP languages [GMW’86]
  - Assuming one-way functions exist
- ZK for all of IP [BGGHKMR’88]
- Everything that can be proven can be proven in zero-knowledge! (Assuming OWF)
ZK Results

- IP and ZK defined [GMR’85]
- ZK for all NP languages [GMW’86]
  - Assuming one-way functions exist
- ZK for all of IP [BGGHKMR’88]
  - Everything that can be proven can be proven in zero-knowledge! (Assuming OWF)
- Variants (known for NP)
ZK Results

- IP and ZK defined [GMR’85]
- ZK for all NP languages [GMW’86]
  - Assuming one-way functions exist
- ZK for all of IP [BGGHKMR’88]
  - Everything that can be proven can be proven in zero-knowledge! (Assuming OWF)
- Variants (known for NP)
  - ZKPoK, Statistical ZK Arguments, Non-Interactive ZK (using a common random string), Witness-Indistinguishable Proofs, ...
ZK Proofs: What for?
ZK Proofs: What for?

Authentication
ZK Proofs: What for?

- Authentication
  - Using ZK Proof of Knowledge
ZK Proofs: What for?

- Authentication
- Using ZK Proof of Knowledge
- Canonical use: As a tool in larger protocols
ZK Proofs: What for?

- Authentication
  - Using ZK Proof of Knowledge
- Canonical use: As a tool in larger protocols
  - To enforce "honest behavior" in protocols
ZK Proofs: What for?

- Authentication
- Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce "honest behavior" in protocols

At each step prove in ZK it was done as prescribed
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me $x_1$ is what you should have sent me now
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me \( x_1 \) is what you should have sent me now
ZK Proofs: What for?

- Authentication
  - Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me $x_1$ is what you should have sent me now

OK
ZK Proofs: What for?

- **Authentication**
  - Using ZK Proof of Knowledge

- **Canonical use: As a tool in larger protocols**
  - To enforce “honest behavior” in protocols
  - At each step prove in ZK it was done as prescribed

Prove to me $x_1$ is what you should have sent me now

OK
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me $x_1$ is what you should have sent me now

OK

Prove $y_1$ is what...
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me $x_1$ is what you should have sent me now

Prove $y_1$ is what...
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me \( x_1 \) is what you should have sent me now

Prove \( y_1 \) is what...

OK

OK
ZK Proofs: What for?

Authentication

Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols

To enforce “honest behavior” in protocols

At each step prove in ZK it was done as prescribed

Prove to me \( x_1 \) is what you should have sent me now

Prove \( y_1 \) is what...

OK

OK
ZK Proofs: What for?

Authentication
Using ZK Proof of Knowledge

Canonical use: As a tool in larger protocols
To enforce “honest behavior” in protocols
At each step prove in ZK it was done as prescribed

Prove $x_1$ is what I should have sent me now
OK

Prove $y_1$ is what...
Prove $x_2$ is what...

OK
Does it fit in?
Does it fit in?

Does the proof stay ZK in the big picture?
Does it fit in?

Does the proof stay ZK in the big picture?

Composition
Does it fit in?

Does the proof stay ZK in the big picture?

Composition

Several issues: auxiliary information from previous runs, concurrency issues, malleability/man-in-the-middle
Does it fit in?

Does the proof stay ZK in the big picture?

Composition

- Several issues: auxiliary information from previous runs, concurrency issues, malleability/man-in-the-middle

In general, to allow composition more complicated protocols
Composition Issues
Multiple executions provide new opportunities for the hacker
Composition Issues

Multiple executions provide new opportunities for the hacker

Play the GM’s against each other
Will not lose against both!
Composition Issues

Multiple executions provide new opportunities for the hacker

Person-in-the-middle attack

Play the GM’s against each other
Will not lose against both!
Composition Issues

- Multiple executions provide new opportunities for the hacker
- Person-in-the-middle attack
- Simulability of a single execution doesn’t imply simulation for multiple executions
Composition Issues

- Multiple executions provide new opportunities for the hacker
- Person-in-the-middle attack
- Simulability of a single execution doesn’t imply simulation for multiple executions
Composition Issues

- Multiple executions provide new opportunities for the hacker
- Person-in-the-middle attack
- Simulability of a single execution doesn’t imply simulation for multiple executions
- Or when run along with other protocols
Universal Composition
Universal Composition

- A security guarantee
Universal Composition

A security guarantee that can be given for a “composed system”
Universal Composition

- A security guarantee
  - that can be given for a “composed system”
  - such that security for each component separately implies security for the entire system
Universal Composition

- A security guarantee
  - that can be given for a “composed system”
  - such that security for each component separately implies security for the entire system
  - and is meaningful! (otherwise, “everything is secure” is composable)
Universal Composition

- A security guarantee
  - that can be given *for a “composed system”*
  - such that security for each component separately implies security for the entire system
  - and is meaningful! (otherwise, “everything is secure” is composable)
- Will use SIM security