Interactive and Zero Knowledge Proofs
Interactive and Zero Knowledge Proofs

Example: Ali Baba's Cave
Interactive and Zero Knowledge Proofs

Example: Ali Baba's Cave

Initially:
Prover and Verifier at the mouth of the cave. Neither can see deep into the cave. From Q a player cannot see either R or S.
**Interactive and Zero Knowledge Proofs**

**Example:** Ali Baba's Cave

Initially:
Prover 🟠 and Verifier 🔴 at the mouth of the cave. Neither can see deep into the cave. From Q a player cannot see either R or S. Prover proves it knows the secret words that will open the door at green line, deep inside the cave, but without telling what they are.
Interactive and Zero Knowledge Proofs

Example: Ali Baba's Cave

Round:
Prover's commitment is to visit \( R \) or \( S \) while verifier waits at \( P \)
Interactive and Zero Knowledge Proofs

Example: Ali Baba's Cave

Round:
Prover's commitment is to visit $R$ or $S$ while verifier waits at $P$.
Verifier's challenge is to walk to $Q$ and ask prover to exit at $R$ or $S$.
Interactive and Zero Knowledge Proofs

Example: Ali Baba's Cave

Round:
Prover's commitment is to visit $R$ or $S$ while verifier waits at $P$
Verifier's challenge is to walk to $Q$ and ask prover to exit at $R$ or $S$
Prover's response is to do as verifier says
Interactive and Zero Knowledge Proofs

Example: Ali Baba's Cave

**Round:**
Prover's commitment is to visit $R$ or $S$ while verifier waits at $P$.
Verifier's challenge is to walk to $Q$ and ask prover to exit at $R$ or $S$.
Prover's response is to do as verifier says.

**Many Rounds:** certain prover does not know or pretty sure it does.
Interactive and Zero Knowledge Proofs

A protocol between two parties in which one party, called the **prover** tries to prove a certain fact to the other party, called the **verifier**. Used for authentication and identification.

**Identification**: a user claims an identity with a username, a process ID, a smart card, etc. Security systems use this identity when determining if the user can access an object.

**Authentication**: the process of proving an identity using given credentials. Examples of credentials:
- a password or PIN (something the prover knows)
- a smart card, CAC, or RSA token (something the prover has)
- biometric information (something that is part of the prover)
Interactive and Zero Knowledge Proofs

A protocol between two parties in which one party, called the prover, tries to prove a certain fact to the other party, called the verifier. Used for authentication and identification.

The fact to prove usually is related to the prover's identity, say the prover's secret key.
Interactive and Zero Knowledge Proofs

A protocol between two parties in which one party, called the **prover** tries to prove a certain fact to the other party, called the **verifier**. Used for authentication and identification.

The fact to prove usually is related to the prover's identity, say the prover's secret key.

The following properties are important:

1. **Completeness** - the verifier always accepts the proof if the fact is true and both parties follow the protocol.
Interactive and Zero Knowledge Proofs

A protocol between two parties in which one party, called the **prover** tries to prove a certain fact to the other party, called the **verifier**. Used for authentication and identification.

The fact to prove usually is related to the prover's identity, say the prover's secret key.

The following properties are important:

1. **Completeness** - the verifier always accepts the proof if the fact is true and both parties follow the protocol.
2. **Soundness** - the verifier always rejects the proof if the fact is false, as long as the verifier follows the protocol.
Interactive and Zero Knowledge Proofs

A protocol between two parties in which one party, called the **prover** tries to prove a certain fact to the other party, called the **verifier**. Used for authentication and identification.

The fact to prove usually is related to the prover's identity, say the prover's secret key.

The following properties are important:

1. **Completeness** - the verifier always accepts the proof if the fact is true and both parties follow the protocol.
2. **Soundness** - the verifier always rejects the proof if the fact is false, as long as the verifier follows the protocol.
3. **Zero-Knowledge** - verifier learns nothing else about the fact being proved from the prover that could not be learned without the prover, regardless of following the protocol. Verifier cannot even prove the fact to anyone later.
Interactive and Zero Knowledge Proofs

How do you know you have a Zero Knowledge proof?
Interactive and Zero Knowledge Proofs

How do you know you have a Zero Knowledge proof?

The verifier can produce a simulation of the transactions even if the prover does not know the fact to be proved. The simulation can be handed to a third party who cannot tell whether the simulation is real or fake.
Interactive and Zero Knowledge Proofs

How do you know you have a Zero Knowledge proof?

The verifier can produce a simulation of the transactions even if the prover does not know the fact to be proved. The simulation can be handed to a third party who cannot tell whether the simulation is real or fake.

How can the verifier do that?
Interactive and Zero Knowledge Proofs

How do you know you have a Zero Knowledge proof?

The verifier can produce a simulation of the transactions even if the prover does not know the fact to be proved. The simulation can be handed to a third party who cannot tell whether the simulation is real or fake.

How can the verifier do that?

The verifier video tapes the transactions and throws out any bad frames and presto the rest looks to anyone like a transaction proving the fact.
Interactive and Zero Knowledge Proofs

A Round - a commitment message from the prover,
   a challenge from the verifier,
   a response to the challenge from the prover.
Interactive and Zero Knowledge Proofs

A Round - a commitment message from the prover,
a challenge from the verifier,
a response to the challenge from the prover.

The protocol may repeat for several rounds. Based on the prover's responses in all the rounds, the verifier decides whether to accept or reject the proof.
Interactive and Zero Knowledge Proofs

How do we know Ali Baba's protocol is a zero-knowledge proof?
Interactive and Zero Knowledge Proofs

How do we know Ali Baba's protocol is a zero-knowledge proof?

Recall we want this: *The proof can be performed efficiently by a simulator that has no idea of what the proof is.*
Interactive and Zero Knowledge Proofs

How do we know Ali Baba's protocol is a zero-knowledge proof?

Recall we want this: *The proof can be performed efficiently by a simulator that has no idea of what the proof is.*

Suppose the prover *does not know* the secret words. Then some of the rounds show the prover unable to find the correct exit. Those rounds are deleted from the video tape.
Interactive and Zero Knowledge Proofs

How do we know Ali Baba's protocol is a zero-knowledge proof?

Recall we want this: *The proof can be performed efficiently by a simulator that has no idea of what the proof is.*

Suppose the prover *does not know* the secret words. Then some of the rounds show the prover unable to find the correct exit. Those rounds are deleted from the video tape.

The result is a sequence of rounds that appear to show the prover *does know* the secret words.
Interactive and Zero Knowledge Proofs

How do we know Ali Baba's protocol is a zero-knowledge proof?

Recall we want this: *The proof can be performed efficiently by a simulator that has no idea of what the proof is.*

Suppose the prover *does not know* the secret words. Then some of the rounds show the prover unable to find the correct exit. Those rounds are deleted from the video tape.

The result is a sequence of rounds that appear to show the prover *does know* the secret words.

Hence a judge looking at the tape cannot be sure whether the prover really knows the secret. Thus, no knowledge concerning the secret can be extracted from the video tape and there is no guaranteed knowledge in the recording of the original protocol.
A mapping of vertices from the left graph to the right graph such that any two vertices in the graph on the left are adjacent iff the mapped vertices in the graph on the right are.
Graph Isomorphism & Zero Knowledge Proofs

It is really hard to determine whether two graphs are isomorphic
It is really hard to determine whether two graphs are isomorphic. But, if someone hands you a vertex mapping, it is easy to check!!!
Graph Isomorphism & Zero Knowledge Proofs

A permutation:

\[
\pi: \begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}
\]

\[
\begin{array}{cccccccc}
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\end{array}
\]
Graph Isomorphism & Zero Knowledge Proofs

A composition of permutations:

\[ \pi: \]

\[
\begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}
\]

\[ \rho: \]

\[
\begin{array}{cccccccccc}
8 & 7 & 6 & 1 & 5 & 2 & 3 & 4 & 9 \\
\end{array}
\]
A composition of permutations is a permutation:

\[ \pi: 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \]

\[ \rho: 8 \quad 7 \quad 6 \quad 1 \quad 5 \quad 2 \quad 3 \quad 4 \quad 9 \]

\[ \rho \circ \pi \]
A permutation has an inverse:

\[
\begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\pi: & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \\
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\pi^{-1}: & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
\end{array}
\]
Graph Isomorphism & Zero Knowledge Proofs

Prover constructs $G_0$ and $\pi$ to get $G_1$. These are made public.

$G_0$: 1 2 3 4 5 6 7 8 9

$G_1$: 2 9 3 1 4 6 8 7 5

$\pi$: 

\[ \begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\end{array} \]
Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$

$G_0$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi$:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G_1$</td>
<td>2</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>$\rho$:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$H$</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>9</td>
</tr>
</tbody>
</table>

$H$ is sent to Verifier
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$

$G_0$:

\[
\begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}
\]

$G_1$:

\[
\begin{array}{cccccccc}
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\end{array}
\]

$\pi$: 

$\rho$: 

$H$:

\[
\begin{array}{cccccccc}
8 & 7 & 6 & 1 & 5 & 2 & 3 & 4 & 9 \\
\end{array}
\]

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_0$ that results in $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$

$G_0$

1 2 3 4 5 6 7 8 9

$G_1$

2 9 3 1 4 6 8 7 5

$\pi$:...

$H$

8 7 6 1 5 2 3 4 9

$\rho$:...

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_0$ that results in $H$

Prover provides $\rho \circ \pi \rightarrow$ applied to $G_0$ gives $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$

$G_0$

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

$G_1$

| 2 | 9 | 3 | 1 | 4 | 6 | 8 | 7 | 5 |

$H$

| 8 | 7 | 6 | 1 | 5 | 2 | 3 | 4 | 9 |

$H$ is sent to Verifier
Verifier asks Prover to provide a permutation from $G_0$ that results in $H$
Prover provides $\rho \circ \pi \rightarrow$ applied to $G_0$ gives $H$
Verifier checks original $H$ against $\rho \circ \pi (G_0)$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes with one-time $\rho$ to get $H$

\[
G_0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \\
\pi:\quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
G_1 \quad 2 \quad 9 \quad 3 \quad 1 \quad 4 \quad 6 \quad 8 \quad 7 \quad 5 \\
\rho:\quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
H \quad 8 \quad 7 \quad 6 \quad 1 \quad 5 \quad 2 \quad 3 \quad 4 \quad 9
\]

$H$ is sent to Verifier
Suppose the Prover chooses $G_I$, permutes with one-time $\rho$ to get $H$

$G_0$: 1 2 3 4 5 6 7 8 9

$G_I$: 2 9 3 1 4 6 8 7 5

$H$: 8 7 6 1 5 2 3 4 9

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_I$ that results in $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$

$G_0 \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \end{array}$

$G_1 \begin{array}{cccccccccc} 2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\ \end{array}$

$H \begin{array}{cccccccccc} 8 & 7 & 6 & 1 & 5 & 2 & 3 & 4 & 9 \\ \end{array}$

$H$ is sent to Verifier
Verifier asks Prover to provide a permutation from $G_1$ that results in $H$
Prover provides $\rho \rightarrow$ applied to $G_1$ gives $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_1$, permutes w/ one-time $\rho$ to get $H$

\[
\begin{align*}
G_0 & \quad 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \\
G_1 & \quad 2 \ 9 \ 3 \ 1 \ 4 \ 6 \ 8 \ 7 \ 5 \\
H & \quad 8 \ 7 \ 6 \ 1 \ 5 \ 2 \ 3 \ 4 \ 9 \\
\end{align*}
\]

$H$ is sent to Verifier
Verifier asks Prover to provide a permutation from $G_1$ that results in $H$
Prover provides $\rho \rightarrow$ applied to $G_1$ gives $H$
Verifier checks original $H$ against $\rho \ (G_1)$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes with one-time $\rho$ to get $H$

\[
G_0 \quad \begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\pi: & & & & & & & & \\
G_1 \quad & 2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\end{array}
\]

\[
\rho: \quad \begin{array}{cccccccc}
8 & 7 & 6 & 1 & 5 & 2 & 3 & 4 & 9 \\
\end{array}
\]

$H$ is sent to Verifier
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes w/ one-time $\rho$ to get $H$

$G_0$
\[
\begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}
\]

$G_1$
\[
\begin{array}{cccccccccc}
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\end{array}
\]

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_1$ that results in $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes w/ one-time $\rho$ to get $H$

\[
G_0 = \begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}
\]

\[
G_1 = \begin{array}{cccccccccc}
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5 \\
\end{array}
\]

$\pi: \ 
\begin{array}{cccccccccc}
\pi(1) & \pi(2) & \pi(3) & \pi(4) & \pi(5) & \pi(6) & \pi(7) & \pi(8) & \pi(9) \\
8 & 7 & 6 & 1 & 5 & 2 & 3 & 4 & 9 \\
\end{array}
$

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_1$ that results in $H$

Prover provides $\rho \circ \pi^{-1} \rightarrow$ applied to $G_1$ gives $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes w/ one-time $\rho$ to get $H$

$G_0$

\[
\begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
\end{array}
\]

$G_1$

\[
\begin{array}{cccccccccc}
2 & 9 & 3 & 1 & 4 & 6 & 8 & 7 & 5
\end{array}
\]

$\pi$: \[
\begin{array}{cccccccccc}
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow
\end{array}
\]

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_1$ that results in $H$

Prover provides $\rho \circ \pi^{-1} \rightarrow$ applied to $G_1$ gives $H$

Verifier checks original $H$ against $\rho \circ \pi^{-1} (G_1)$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes with one-time $\rho$ to get $H$

$G_0$

1 2 3 4 5 6 7 8 9

$G_1$

2 9 3 1 4 6 8 7 5

$H$ is sent to Verifier
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes w/ one-time $\rho$ to get $H$

$G_0$ 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9
$G_1$ 2 9 3 1 4 6 8 7 5 8 7 6 1 5 2 3 4 9

$H$ is sent to Verifier
Verifier asks Prover to provide a permutation from $G_0$ that results in $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes w/ one-time $\rho$ to get $H$

\[
\begin{align*}
G_0 & \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \\
\pi: & \\
G_1 & \quad 2 \quad 9 \quad 3 \quad 1 \quad 4 \quad 6 \quad 8 \quad 7 \quad 5 \\
\rho: & \\
8 \quad 7 \quad 6 \quad 1 \quad 5 \quad 2 \quad 3 \quad 4 \quad 9
\end{align*}
\]

$H$ is sent to Verifier
Verifier asks Prover to provide a permutation from $G_0$ that results in $H$
Prover provides $\rho \rightarrow$ applied to $G_0$ gives $H$
Graph Isomorphism & Zero Knowledge Proofs

Suppose the Prover chooses $G_0$, permutes w/ one-time $\rho$ to get $H$

$G_0$

1 2 3 4 5 6 7 8 9

$G_1$

2 9 3 1 4 6 8 7 5

$\pi$: 

$\rho$: 

$H$ is sent to Verifier

Verifier asks Prover to provide a permutation from $G_0$ that results in $H$

Prover provides $\rho \rightarrow$ applied to $G_0$ gives $H$

Verifier checks original $H$ against $\rho$ ($G_0$)
Graph Isomorphism - Zero Knowledge Proof

Prover

Verifier
Graph Isomorphism - Zero Knowledge Proof

Generate two isomorphic graphs, $G_0$ and $G_1$ of $n$ vertices, where $G_1 = \pi(G_0)$. Keep $\pi$ secret, publish the graphs.
Graph Isomorphism - Zero Knowledge Proof

Protocol:

Prover: Generate 2nd perm $\rho$, compute $H=\rho(G_e)$, select $e \in \{0,1\}$
Graph Isomorphism - Zero Knowledge Proof

Protocol:

Prover: Generate 2nd perm \( \rho \), compute \( H = \rho(G_e) \), select \( e \in \{0,1\} \)

Verifier: Select \( e' \in \{0,1\} \), ask prover to prove \( H \) isomorphic to \( G_{e'} \)
Graph Isomorphism - Zero Knowledge Proof

**Protocol:**

**Prover:** Generate 2nd perm $\rho$, compute $H = \rho(G_e)$, select $e \in \{0,1\}$

**Verifier:** Select $e' \in \{0,1\}$, ask prover to prove $H$ isomorphic to $G_{e'}$

**Prover:** Compute $\sigma = \begin{cases} 
\rho & \text{if } e' = e \\
\rho \circ \pi^{-1} & \text{if } e' = 1 \text{ and } e = 0 \\
\rho \circ \pi & \text{if } e' = 0 \text{ and } e = 1 
\end{cases}$
Graph Isomorphism - Zero Knowledge Proof

Protocol:

Prover: Generate 2nd perm $\rho$, compute $H = \rho(G_e)$, select $e \in \{0,1\}$

Verifier: Select $e' \in \{0,1\}$, ask prover to prove $H$ isomorphic to $G_{e'}$

Prover: Compute $\sigma = \begin{cases} 
\rho & \text{if } e' = e \\
\rho \circ \pi^{-1} & \text{if } e' = 1 \text{ and } e = 0 \\
\rho \circ \pi & \text{if } e' = 0 \text{ and } e = 1 
\end{cases}$

Verifier: Checks $\sigma(G_{e'}) = H$
Graph Isomorphism - Zero Knowledge Proof

Protocol:

**Impersonator:** Generate $\rho$, compute $H = \rho(G_e)$, select $e \in \{0, 1\}$
Graph Isomorphism - Zero Knowledge Proof

**Protocol:**

**Impersonator:** Generate $\rho$, compute $H = \rho(G_e)$, select $e \in \{0,1\}$

**Verifier:** Select $e' \in \{0,1\}$, ask to prove $H$ isomorphic to $G_{e'}$
Graph Isomorphism - Zero Knowledge Proof

Protocol:

**Impersonator:** Generate $\rho$, compute $H=\rho(G_e)$, select $e \in \{0,1\}$

**Verifier:** Select $e' \in \{0,1\}$, ask to prove $H$ isomorphic to $G_{e'}$

**Impersonator:** Cannot compute $\sigma$ if $e' \neq e$, does not know $\pi$
Graph Isomorphism - Zero Knowledge Proof

**Protocol:**

**Impersonator:** Generate $\rho$, compute $H = \rho(G_e)$, select $e \in \{0,1\}$

**Verifier:** Select $e' \in \{0,1\}$, ask to prove $H$ is isomorphic to $G_{e'}$

**Impersonator:** Cannot compute $\sigma$ if $e' \neq e$, does not know $\pi$

$\rho$ is always changed so impersonator can't determine $\pi$ from eavesdropping
Fiat-Shamir Zero Knowledge Proof

Based on difficulty of computing square roots mod a composite $n$
Fiat-Shamir Zero Knowledge Proof

Based on difficulty of computing square roots mod a composite $n$
Given two large primes $p, q$ and $n = p \times q$, computing $\sqrt{x} \mod n$
is very hard without knowing $p, q$
Fiat-Shamir Zero Knowledge Proof

Based on difficulty of computing square roots mod a composite $n$

Given two large primes $p, q$ and $n=p*q$, computing $\sqrt{x} \mod n$
is very hard without knowing $p, q$

But there exist efficient algorithms for computing square roots
modulo a prime number, and therefore $\sqrt{x} \mod n$ can be
computed efficiently if $p$ and $q$ are known
by the Chinese Remainder Theorem:

Example: find $\sqrt{5} \mod (11*19)$

Note: $\sqrt{5} \mod 11 = 4$ or $7$

$\sqrt{5} \mod 19 = 9$ or $10$

Ans: $9*7*11 \mod 11*19 + 4*7*19 \mod 11*19 = 180$

Chk: $180*180 \mod 11*19 = 5$
Fiat-Shamir Zero Knowledge Proof

Prover

Trusted Party
Fiat-Shamir Zero Knowledge Proof

Prover

Trusted Party

$n = p*q$
Fiat-Shamir Zero Knowledge Proof

Prover

1 = \text{gcd}(n, S)
V = S^2 \mod n

Trusted Party

n = pq

n = p \ast q

Fiat-Shamir Zero Knowledge Proof

Prover

Verifier

$S$

$V$
Fiat-Shamir Zero Knowledge Proof

Prover chooses random $r$, sends $r^*r \mod n$
Fiat-Shamir Zero Knowledge Proof

Prover chooses random $r$, sends $r^* r \mod n$

Verifier chooses 1 or 0 and sends it to prover

$e \in \{1,0\}$
Fiat-Shamir Zero Knowledge Proof

Prover chooses random \( r \), sends \( r^* r \mod n \)

Verifier chooses 1 or 0 and sends it to prover

Prover sends \( r^* S^e \mod n \) to verifier

\[ a = r^* S^e \mod n \]
Fiat-Shamir Zero Knowledge Proof

Prover chooses random $r$, sends $r^* r \mod n$
Verifier chooses 1 or 0 and sends it to prover
Prover sends $r^* S^e \mod n$ to verifier
Verifier checks $a^* a$ against $V^e * r^* r \mod n$
Fiat-Shamir Zero Knowledge Proof

$S$

$V$

$V^e * r * r \mod n = r * r \mod n$ if $e=0$ or
$S * S * r * r \mod n$ if $e=1$

Verifier checks $a * a$ against $V^e * r * r \mod n$

If prover knows $S$, then verifier's test always succeeds
Otherwise it fails half the time

$a = r * S^e \mod n$
Feige-Fiat-Shamir Zero Knowledge Proof

Based on difficulty of computing square roots mod a composite $n$
Given two large primes $p, q$ and $n=p\cdot q$, computing $\sqrt{x} \mod n$ is very hard without knowing $p, q$

But there exist efficient algorithms for computing square roots modulo a prime number, and therefore $\sqrt{x} \mod n$ can be computed efficiently if $p$ and $q$ are known
Feige-Fiat-Shamir Zero Knowledge Proof

Prover

Verifier
Feige-Fiat-Shamir Zero Knowledge Proof

\[ p, q, S \]
\[ n = p \times q \]
\[ V = S \times S \mod n \]
Feige-Fiat-Shamir Zero Knowledge Proof

**Protocol:**

**Prover:** Generate random $r, s \in \{-1,1\}$, send $x = s^*r*r \mod n$
Feige-Fiat-Shamir Zero Knowledge Proof

Protocol:

**Prover:** Generate random $r$, $s \in \{-1,1\}$, send $x = s^r \cdot r \mod n$

**Verifier:** Select $e \in \{0,1\}$
Feige-Fiat-Shamir Zero Knowledge Proof

**Protocol:**

**Prover:** Generate random $r$, $s \in \{-1, 1\}$, send $x = s*r*r \mod n$

**Verifier:** Select $e \in \{0,1\}$

**Prover:** Send $y = r*S^e \mod n$
Feige-Fiat-Shamir Zero Knowledge Proof

\[ p, q, S \]
\[ n = p \times q \]
\[ V = S \times S \mod n \]

**Protocol:**

**Prover:** Generate random \( r, s \in \{-1, 1\} \), send \( x = s \times r \times r \mod n \)

**Verifier:** Select \( e \in \{0, 1\} \)

**Prover:** Send \( y = r \times S^e \mod n \)

**Verifier:** Checks \( y \times y = \mp x \times V^e \mod n \)
Feige-Fiat-Shamir Zero Knowledge Proof

$p, q, S$
$n = p \times q$

\[ V = S \times S \mod n \]

Security:
Attacker does not know $S$. But if it knew what $e$ the verifier would send, it could pick a random $y$, calculate $x = y^2 \times V^e \mod n$ and send $x$ to the verifier. When the verifier sends $e$, the attacker returns $y$. Squaring this will match what was sent before. With probability $1/2$ the attacker will make the wrong guess for $e$. 
Parallel Zero Knowledge Protocols

Send $m$ commitments in one message
Parallel Zero Knowledge Protocols

{c(1), c(2),...,c(m)}

Send $m$ commitments in one message
But cannot simulate!!!
Cannot edit the Tape!!!
Parallel Zero Knowledge Protocols

{c(1), c(2),..., c(m)}

Prover → Verifier

Send \( m \) commitments in one message
But cannot simulate!!!
Cannot edit the Tape!!!
Are we screwed???
Security Problems

Prover × public key Attacker → Verifier
Security Problems

Verifier checks with the Trust Center whenever it needs the Prover's public key.

Prover

Verifier

Trust Center with Key Dictionary

public key
Security Problems

**Even better:** need use Trust Center only for key generation!

**Here's How:** Trust Center does the following one time:
- Generates primes $p$, $q$, and computes $n=p*q$
- Publishes $n$, keeps $p$, $q$ secret
- Defines and publishes a one-way hash function $f$

**Then:** a Prover visits the Trust Center for a Zero-Knowledge ID
Security Problems

At the Trust Center:

Prover's ID info \[ f \] Prover's public key \( v \)
Security Problems

At the Trust Center:

Prover's ID info $\rightarrow f \rightarrow$ Prover's public key $v$

\[ \sqrt{v} \mod p \]

\[ \sqrt{v} \mod q \]

Prover's private key $s = \sqrt{v} \mod n$
Security Problems

At the Trust Center:

Prover's ID info

$f$

Prover's public key $v$

$\sqrt{v} \mod p$

Prover's private key $s = \sqrt{v} \mod n$

$\sqrt{v} \mod q$

Prover's Certified Data

$\sqrt{v} \mod n$

Prover's ID info

To the Prover
Security Problems

At the Verifier:

Prover's ID info $\rightarrow f \rightarrow$ Prover's public key $\nu$

Then run the Zero-Knowledge Authentication Scheme

Attacker can spoof the Prover's ID info but will not know the Prover's secret and will not be able to convince the Verifier that it is the Prover