ZERO-KNOWLEDGE PROOFS
WHAT PROPERTIES SHOULD AN INTERACTIVE PROOF SYSTEM HAVE?
Alice$(G_1, G_2, \rho)$

$\pi \leftarrow S_n$

$H \leftarrow \pi(G_1)$

$\sigma \leftarrow \begin{cases} 
\pi^{-1} & \text{IF } c=1 \\
\rho \cdot \pi^{-1} & \text{O.w.}
\end{cases}$

Bob$(G_1, G_2)$

$c \leftarrow \{1, 2\}$

$G_c \overset{?}{=} \sigma(H)$
QUADRATIC EQUATIONS

\[ N = pq \]
\[ x^2 - 7 = 0 \mod N \]

PROVE THAT EQUATION HAS A SOLUTION
WITHOUT REVEALING THE SOLUTION!

**Alice** \((N, 7, x)\)

\[ r \leftarrow \mathbb{Z}_N^* \]
\[ t \leftarrow r^2 \cdot 7 \]

\[ s \leftarrow \begin{cases} r & c = 0 \\ rx & c = 1 \end{cases} \]

**Bob** \((N, 7)\)

\[ c \leftarrow \{0, 1\} \]

CHECK \( s^2 \cdot 7^{1-c} \equiv t \)
Completeness

“IF A STATEMENT IS TRUE, THEN THE PROVER SHOULD ALWAYS BE ABLE TO CONVINCE THE VERIFIER THAT IT IS SO.”

WHAT DOES IT MEAN TO BE TRUE?

LANGUAGE $L$ IS A SET OF STRINGS

$L = \{p \mid p \text{ is prime}\}$

$L = \{(G_1, G_2) \mid G_1 \sim G_2\}$

$L = \{(N, y) \mid N = pq \text{ and } \exists x \text{ s.t. } x^2 - y^2 = 0 \mod n\}$

$x$ IS TRUE RELATIVE TO $L$ IF $x \in L$
COMPLETENESS

FOR A LANGUAGE $L$, AND FOR ALL $x \in L$

$$\langle P(x, w), V(x) \rangle = 1$$

“INTERACTION WITH $V$ ACCEPTING THE PROOF”
COMPLETENESS

FOR A LANGUAGE $L$, AND FOR ALL $x \in L$

$$\Pr[\langle P(x, w), V(x) \rangle = 1]$$

“INTERACTION WITH $V$ accepting THE PROOF”
Completeness

For a language $L$, and for all $x \in L$

$$\mathbb{P}_R[\langle P(x, w), V(x) \rangle = 1] = 1$$

"Interaction with $V$ accepting the proof"
COMPLETENESS

For a language $L$, and for all $x \in L$

$$\Pr[\langle P(x, w), V(x) \rangle = 1] = 1 - \epsilon(|x|)$$

"Interaction with $V$ accepting the proof"
SOUNDNESS

“IF A STATEMENT IS FALSE, THEN THE PROVER SHOULD NEVER BE ABLE TO CONVINCE THE VERIFIER THAT IT IS SO.”
SOUNDNESS

“IF A STATEMENT IS FALSE, THEN THE PROVER SHOULD NEVER BE ABLE TO CONVINCE THE VERIFIER THAT IT IS SO.”

FOR A LANGUAGE $L$, AND FOR ALL $x \notin L$

$\in$
SOUNDNESS

“IF A STATEMENT IS FALSE, THEN THE PROVER SHOULD NEVER BE ABLE TO CONVINCE THE VERIFIER THAT IT IS SO.”

FOR A LANGUAGE $L$, AND FOR ALL $x \not\in L$

$$\Pr[ \left< P(x, w), V(x) \right> = 1 ] < \epsilon(|x|)$$

WHERE $\epsilon$ IS A NEGLIGIBLE FUNCTION
\( Alice(G_1, G_2, \rho) \)

\( Bob(G_1, G_2) \)

= 1
Alice \((G_1, G_2, \rho)\)

CHECK \(G_1 \cong \rho(G_2)\)

Bob \((G_1, G_2)\)

\[= 1\]
Alice$(G_1, G_2, \rho)$

CHECK $G_1 \cong \rho(G_2)$

IF YES, SEND “IT IS TRUE”

Bob$(G_1, G_2)$

$= 1$
$\text{Alice}(G_1, G_2, \rho)$

CHECK $G_1 \stackrel{?}{=} \rho(G_2)$

IF YES, SEND “IT IS TRUE”

IF NO, SEND “IT IS NOT TRUE”

$= 1$
Alice\((G_1, G_2, \rho)\)

CHECK \(G_1 \stackrel{?}{=} \rho(G_2)\)

IF YES, SEND “IT IS TRUE”

IF NO, SEND “IT IS NOT TRUE”

Bob\((G_1, G_2)\)

\[= 1\]
Alice($G_1, G_2, \rho$)

CHECK $G_1 \overset{?}{=} \rho(G_2)$

IF YES, SEND “IT IS TRUE”

IF NO, SEND “IT IS NOT TRUE”

Bob($G_1, G_2$)

FOR A LANGUAGE $L$, AND FOR ALL $x \notin L$

$$\Pr[\langle P(x, w), V(x) \rangle = 1 ] < \epsilon(|x|)$$

WHERE $\epsilon$ IS A NEGLIGIBLE FUNCTION
SOUNDNESS

“IF A STATEMENT IS FALSE, THEN NO PROVER SHOULD EVER BE ABLE TO CONVINCE THE VERIFIER THAT IT IS SO.”

FOR A LANGUAGE $L$, AND FOR ALL $x \notin L$ \( \forall P^* \)

\[
\Pr[ \langle P^*(x), V(x) \rangle = 1 ] < \epsilon(|x|)
\]

WHERE $\epsilon$ IS A NEGLIGIBLE FUNCTION
INTERACTIVE PROTOCOLS ARE POWERFUL
CAN YOU PROVE THAT THIS EQUATION HAS NO SOLUTIONS:

\[ x^2 - 3 = 0 \mod 77 \]
PROVE THIS EQUATION HAS NO SOLUTIONS

\[ x^2 - 3 = 0 \mod 77 \]

**Alice** \((N, p, q, y = 3)\)

**Bob** \((N, 3)\)

**IF** \(t\) **IS A SQUARE**

SEND \(s = 0\)

**IF NOT**

SEND \(s = 1\)

\[ r \leftarrow Z_N^* \]

\[ b \leftarrow \{0, 1\} \]

\[ t \leftarrow r^2 \cdot 3^b \]

ACCEPT IF \(s = b\)
PROVE THIS EQUATION HAS NO SOLUTIONS

\[ x^2 - 3 = 0 \mod 77 \]

IF THERE ARE NO SOLUTIONS, THEN CHALLENGES ARE

\[ b=1 \]

\[ b=0 \]
PROVE THIS EQUATION HAS NO SOLUTIONS

\[ x^2 - 3 = 0 \mod 77 \]

IF THERE ARE SOLUTIONS, THEN CHALLENGES ARE

\[ b = 0 \]

PROVER IS CAUGHT 1/2 TIME
PROVE THIS EQUATION HAS NO SOLUTIONS

\[ x^2 - 3 = 0 \mod 77 \]

**Alice** \((N, p, q, y = 3)\)

**Bob** \((N, 3)\)

\(r \leftarrow \mathbb{Z}_N^*\)

\(b \leftarrow \{0, 1\}\)

\(t \leftarrow r^2 \cdot 3^b\)

\[ t \]

**If** \(t\) **is a square**

**Send** \(s=1\)

**If not**

**Send** \(s=0\)

**Accept if** \(s=b\)

**But is the protocol zero-knowledge?**
ZERO-KNOWLEDGE

“AFTER SEEING A ZERO-KNOWLEDGE PROOF, A VERIFIER SHOULD BE UNABLE TO ACCOMPLISH ANY NEW TASKS.”
by interacting with the prover, the verifier was convinced the statement was true but the proof itself.... the verifier could have generated them himself!
HOW TO GENERATE A “TRANSCRIPT” OF A PROOF:

FIRST PICK $c$
THEN PICK $\sigma$
THEN SET $H = \sigma^{-1}(G_c)$
OUTPUT $(H, c, \sigma)$
“After seeing a zero-knowledge proof, a verifier should be unable to accomplish any new tasks.”

There exists a p.p.t. machine $S$ s.t. for all $x \in L$

$$\text{View}_V(\langle P(x, w), V(x) \rangle) \approx S(x)$$

“Honest verifier zero-knowledge”
Alice \((G_1, G_2, \rho)\)

\[\pi \leftarrow S_n\]

\[H \leftarrow \pi(G_1)\]

\[\sigma \leftarrow \begin{cases} 
\pi^{-1} & \text{IF } c=1 \\
\rho \cdot \pi^{-1} & \text{IF } c=2 \\
\rho & \text{OTHERWISE}
\end{cases}\]

Bob \((G_1, G_2)\)

\[c \leftarrow \{1, 2\}\]

\[G_c \overset{?}{=} \sigma(H)\]
“AFTER SEEING A ZERO-KNOWLEDGE PROOF, A VERIFIER SHOULD BE UNABLE TO ACCOMPLISH ANY NEW TASKS.”

FOR ALL VERIFIERS $V^*$
THERE EXISTS A P.P.T. MACHINE $S^O$ S.T.
FOR ALL $x \in L$

$\text{View}_{V^*}(\langle P(x, w), V^*(x) \rangle) \approx S^{V^*}(x)$
PROVE THIS EQUATION HAS NO SOLUTIONS

\[ x^2 - 3 = 0 \mod 77 \]

**Alice** \((N, p, q, y = 3)\)

**Bob** \((N, 3)\)

- If \(t\) is a square
  - Send \(s = 1\)
- If not
  - Send \(s = 0\)

\[ t \leftarrow r^2 \cdot 3^b \]

Accept if \(s = b\)

**So is this protocol zero-knowledge?**
PROVE THIS EQUATION HAS NO SOLUTIONS

\[ x^2 - 3 = 0 \mod 77 \]

\[ \textbf{Alice} (N, p, q, y = 3) \]

**IF** \( t \) **IS A SQUARE**

**SEND** \( s = 1 \)

**IF NOT**

**SEND** \( s = 0 \)

\[ \textbf{Bob} (N, 3) \]

\( r \leftarrow Z^*_N \)

\( b \leftarrow \{0, 1\} \)

\( t \leftarrow r^2 \cdot 3^b \)

**ACCEPT** **IF** \( s = b \)

SO IS THIS PROTOCOL ZERO-KNOWLEDGE?

**NO. BUT CAN BE FIXED IN TWO WAYS...**
GRAPH ISO
QUADRATIC EQUATIONS

LET'S BUILD A ZK PROOF FOR MORE GENERAL PROBLEMS.
3-COLORING OF A GRAPH

NP-COMPLETE
3-COLORING OF A GRAPH

NP-COMPLETE
ZK-PROOF OF 3-COLORABILITY

\[ \text{Alice}(G, C) \quad \text{Bob}(G) \]
ZK-PROOF OF 3-COLORABILITY

\[ \text{Alice}(G, C) \quad \text{Bob}(G) \]
ZK-PROOF OF 3-COLORABILITY

Alice\( (G,C) \)  
PICK A COLOR PERM  
Bob\( (G) \)
ZK-PROOF OF 3-COLORABILITY

\[ \text{Alice}(G, C) \]

Pick a color perm

\[ \text{Bob}(G) \]

Color the graph with new perm
ZK-PROOF OF 3-COLORABILITY

\[ \text{Alice}(G, C) \]

PICK A COLOR PERM

COLOR THE GRAPH

PLACE CUPS OVER NODES

\[ \text{Bob}(G) \]
ZK-PROOF OF 3-COLORABILITY

Alice(G, C)
- Pick a color perm
- Color the graph
- Place cups over nodes

Bob(G)
ZK-PROOF OF 3-COLORABILITY

Alice\((G, C)\)

- **PICK A COLOR PERM**
- **COLOR THE GRAPH**
- **PLACE CUPS OVER NODES**

Bob\((G)\)

- **PICK A RANDOM EDGE**
**ZK-PROOF OF 3-COLORABILITY**

*Alice(G, C)*

- Pick a color perm
- Color the graph
- Place cups over nodes

*Bob(G)*

- Pick a random edge
- Reveals chosen edge
zk-proof of 3-colorability

Alice \((G, C)\)

- Pick a color perm
  - Color the graph
  - Place cups over nodes
  - Reveals chosen edge

Bob \((G)\)

- Pick a random edge
  - Check colors
ZK-PROOF OF 3-COLORABILITY

Alice(G, C)
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

Bob(G)
- Pick a random edge
- Check colors

Completeness?
ZK-PROOF OF 3-COLORABILITY

\textbf{Alice}(G,C)
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

\textbf{Bob}(G)
- Pick a random edge
- Check colors

\textbf{Soundness?} \quad G \not\in 3\text{-col}
ZK-PROOF OF 3-COLORABILITY

\[ \text{Alice}(G, C) \]
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

\[ \text{Bob}(G) \]
- Pick a random edge
- Check colors

**SOUNDNESS?** \( G \not\in 3\text{-col} \)
\[ \frac{1}{\# \text{ edges}} = \frac{1}{m} \]
ZK-PROOF OF 3-COLORABILITY

Alice(G, C)
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

Bob(G)
- Pick a random edge
- Check colors

Soundness? $G \not\in 3$-col

\[
\frac{1}{\# \text{ edges}} = \frac{1}{m}
\]

Repeat $m^2$ times
ZK-PROOF OF 3-COLORABILITY

\[ A(d, G) \]
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

\[ B(d, G) \]
- Pick a random edge
- Check colors

\text{SOUNDNESS?} \quad G \not\in 3\text{-col} \quad \frac{1}{\# \text{ edges}} = \frac{1}{m}

\text{Repeat} \quad m^2 \text{ times} \quad \left(1 - \frac{1}{m}\right)^{m^2}
ZK-PROOF OF 3-COLORABILITY

Alice($G, C$)
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

Bob($G$)
- Pick a random edge
- Check colors

Soundness? $G \not\in 3$-col

Repeat $m^2$ times

$$\left(1 - \frac{1}{m}\right)^{m^2} \sim \left(\frac{1}{e}\right)^m$$

$$\frac{1}{\# \text{edges}} = \frac{1}{m}$$
ZK-PROOF OF 3-COLORABILITY

Alice\((G, C)\)
- Pick a color perm
- Color the graph
- Place cups over nodes
- Reveals chosen edge

Bob\((G)\)
- Pick a random edge
- Check colors

ZERO-KNOWLEDGE?
SIMULATOR FOR 3-COL

$\mathcal{S}(G)$

$\mathcal{B}ob^*(G)$
SIMULATOR FOR 3-COL

$S(G)$

$\text{Bob}^*(G)$
SIMULATOR FOR 3-COL

$S(G)$

Bob*(G)

COLOR A RANDOM EDGE $\epsilon$
SIMULATOR FOR 3-COL

\[ S(G) \]

\[ Bob^*(G) \]

COLOR A RANDOM EDGE \( \epsilon \)
PLACE CUPS OVER NODES
SIMULATOR FOR 3-COL

Bob*(G)

S(G)

COLOR A RANDOM EDGE $\ell$
PLACE CUPS OVER NODES
ASK BOB FOR CHALLENGE $c$
$\mathcal{S}(G)$

$\mathcal{B}_a^*(G)$

COLOR A RANDOM EDGE $\epsilon$
PLACE CUPS OVER NODES
ASK BOB FOR CHALLENGE $c$
SUCCESS IF $c = \epsilon$
SIMULATOR FOR 3-COL

Bob^*(G)
S(G)

COLOR A RANDOM EDGE \( e \)
PLACE CUPS OVER NODES
ASK BOB FOR CHALLENGE \( c \)
SUCCESS IF \( c=e \)
ELSE REPEAT
SIMULATOR FOR 3-COL

Bob\(^*(G)\)

\(\Sigma(G)\)

COLOR A RANDOM EDGE \(e\)
PLACE CUPS OVER NODES
ASK BOB FOR CHALLENGE \(c\)
SUCCESS IF \(c = e\)
ELSE REPEAT

SUCCEEDS WITH PR
SIMULATOR FOR 3-COL

$\mathcal{S}(G)$

$\mathcal{B}(G)$

COLOR A RANDOM EDGE $\epsilon$
PLACE CUPS OVER NODES
ASK BOB FOR CHALLENGE $c$
SUCCESS IF $c = \epsilon$
ELSE REPEAT

SUCCEEDS WITH PR $\frac{1}{m}$
SIMULATOR FOR 3-COL

Bob∗(G)
S(G)

COLOR A RANDOM EDGE e
PLACE CUPS OVER NODES
ASK BOB FOR CHALLENGE c
SUCCESS IF c=ε
ELSE REPEAT

SUCCEEDS WITH PR \( \frac{1}{m} \)

EXPECTED # ITERATIONS: m
SUDOKO PUZZLE

BEFORE I START, CAN YOU PROVE THAT IT HAS A SOLUTION?
Before I start, can you prove that it has a solution?

YES. SUDOKO IS IN NP!
**SUDOKO PUZZLE**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>9</td>
<td></td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td>7</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td></td>
<td>9</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Before I start, can you prove that it has a solution?

Yes. Sudoko is in NP!

Yes. There is even an efficient way.
<table>
<thead>
<tr>
<th></th>
<th>6</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td>9</td>
<td></td>
<td>2</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td>3</td>
<td>6</td>
<td></td>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>6</td>
<td>9</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
SUDOKO PUZZLE

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>6</th>
<th></th>
<th></th>
<th>4</th>
<th></th>
<th></th>
<th>3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>8</td>
<td></td>
<td>3</td>
<td>4</td>
<td></td>
<td>1</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td></td>
<td>3</td>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>6</td>
<td></td>
<td></td>
<td>9</td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>4</td>
<td>3</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

# SUDOKO PUZZLE

```
<table>
<thead>
<tr>
<th></th>
<th></th>
<th>6</th>
<th></th>
<th></th>
<th>9</th>
<th></th>
<th>4</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td>3</td>
<td>9</td>
<td></td>
<td>2</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td></td>
<td>4</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>8</th>
<th></th>
<th></th>
<th>1</th>
<th></th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>6</td>
<td>3</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>7</td>
<td>6</td>
<td></td>
<td>9</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td></td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>2</th>
<th>5</th>
<th></th>
<th>1</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```

SUDOKO PUZZLE

- Each row, column, and 3x3 box must contain the digits 1 to 9.

- The solved puzzle will have each digit from 1 to 9 appearing exactly once in each row, column, and 3x3 box.
SUDOKO PUZZLE

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>4</td>
<td>9</td>
<td>4</td>
<td>3</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{Sudoko Puzzle} \]
# SUDOKO PUZZLE

```
<table>
<thead>
<tr>
<th></th>
<th></th>
<th>6</th>
<th></th>
<th></th>
<th></th>
<th>4</th>
<th>3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td>2</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td></td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>6</td>
<td></td>
<td>9</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

This is a standard 9x9 Sudoku puzzle grid. Each row, column, and 3x3 box must contain the digits 1 through 9 exactly once.
<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>6</td>
</tr>
</tbody>
</table>
### SUDOKO PUZZLE

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>8</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>6</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>1</td>
<td>6</td>
<td>9</td>
<td>2</td>
<td>7</td>
<td>8</td>
<td>3</td>
</tr>
</tbody>
</table>

**SHUFFLE**
### SUDOKO PUZZLE

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CHECK EACH PILE HAS ALL 9**
DO THE SAME FOR THE ROWS
AND EACH SUB-SQUARE
HOW TO MAKE THIS PROTOCOL INTERNET-READY?

WHAT FUNCTION DID THE CUPS PLAY?

(binding) prover could not change color

(hiding) verifier could not see color
COMMITMENT SCHEMES

Sender

Sender commits to a bit
Sender cannot change her mind
Receiver cannot learn the bit

Receiver

Sender can open commitment
COMMITMENT SCHEMES

TWO PROTOCOLS: COMMIT() OPEN()

\((c, s) \leftarrow \text{COMMIT}(b; r)\)

***Sender*** \(c\) ***Receiver***

**OPEN**

***Sender*** \(b, s\) ***Receiver***