MIT 6.875 & Berkeley CS276

Foundations of Cryptography
Lecture 16
Today:
Non-Interactive Zero-Knowledge (NIZK)

In Two Days:
An Application of NIZK
NP Proofs

For the NP-complete problem of graph 3-coloring

Prover $P$ has a witness, the 3-coloring of $G$

Verifier $V$ checks:
(a) only 3 colors are used &
(b) any two vertices connected by an edge are colored differently.
Zero-Knowledge (Interactive) Proof

Because NP proofs reveal too much

Commitments

$e \leftarrow E$
Zero-Knowledge (Interactive) Proof

Because NP proofs reveal too much

1. Completeness: For every $G \in 3\text{COL}$, $V$ accepts $P$’s proof.

2. Soundness: For every $G \notin 3\text{COL}$ and any cheating $P^\ast$, $V$ rejects $P^\ast$’s proof with probability $\geq 1 - \text{neg}(n)$

3. Zero Knowledge: For every cheating $V^\ast$, there is a PPT simulator $S$ such that for every $G \in 3\text{COL}$, $S$ simulates the view of $V^\ast$. 
TODAY:

Can we make proofs non-interactive again?

Why?
1. $V$ does not need to be online during the proof process.
2. Proofs are not ephemeral, can stay into the future.
TODAY:

Can we make proofs non-interactive again?

YES, WE CAN!
Non-Interactive ZK is Impossible

Suppose there were an NIZK proof system for 3COL.

Graph G

Step 1. When G is in 3COL, V accepts the proof $\pi$. (Completeness)
Non-Interactive ZK is Impossible

Suppose there were an NIZK proof system for 3COL.

Step 2. PPT Simulator S, given only G in 3COL, produces an indistinguishable proof $\tilde{\pi}$ (Zero Knowledge).

In particular, V accepts $\tilde{\pi}$. 
Non-Interactive ZK is Impossible

Suppose there were an NIZK proof system for 3COL.

Step 3. Imagine running the Simulator $S$ on a $G \notin 3COL$. It produces a proof $\tilde{\pi}$ which the verifier still accepts!

(Why?! Because $S$ and $V$ are PPT. They together cannot tell if the input graph is 3COL or not)
Non-Interactive ZK is Impossible

Suppose there were an NIZK proof system for 3COL.

Step 4. Therefore, $S$ is a cheating prover!

Produces a proof for a $G \notin 3\text{COL}$ that the verifier nevertheless accepts.

Ergo, the proof system is NOT SOUND!
THE END

Or, is it?
Enter: The Common Random String

CRS  010111000101010010

Graph $G$

$\pi$

$P \rightarrow V$
Enter: The Common Reference String

\[ CRS \gets D \]

(e.g., CRS = product of two primes)
1. **Completeness:** For every $G \in 3\text{COL}$, $V$ accepts P’s proof.

2. **Soundness:** For every $G \notin 3\text{COL}$ and any “proof” $\pi^*$, $V(CRS, \pi^*)$ accepts with probability $\leq \text{neg}(n)$
3. Zero Knowledge: There is a PPT simulator S such that for every $G \in 3\text{COL}$, $S$ simulates the view of the verifier $V$.

$$S(G) \approx (\text{CRS} \leftarrow D, \pi \leftarrow P(G, \text{colors}))$$
3. Zero Knowledge: There is a PPT simulator $S$ such that for every $x \in L$ and witness $w$, $S$ simulates the view of the verifier $V$.

$$S(x) \approx (CRS \leftarrow D, \pi \leftarrow P(x, w))$$
HOW TO CONSTRUCT NIZK IN THE CRS MODEL

1. Blum-Feldman-Micali'88 (*quadratic residuosity*)

2. Feige-Lapidot-Shamir’90 (*factoring*)

3. Groth-Ostrovsky-Sahai’06 (*bilinear maps*)

4. Canetti-Chen-Holmgren-Lombardi-Rothblum²-Wichs’19 and Peikert-Shiehian’19 (*learning with errors*)
HOW TO CONSTRUCT NIZK IN THE CRS MODEL

Step 1. **Review** our number theory hammers & polish them.

Step 2. **Construct** NIZK for a special NP language, namely quadratic non-residuosity.

Step 3. **Bootstrap** to NIZK for 3SAT, an NP-complete language.
Quadratic Residuosity

Let $N = pq$ be a product of two large primes.
Quadratic Residuosity

Let $N = pq$ be a product of two large primes.

$Z_N^*$ divides $\mathcal{J}ac$ evenly unless $N$ is a perfect square.
Quadratic Residuosity

Let $N = pq$ be a product of two large primes.

$\mathbb{Z}_N^*$

\[
\begin{aligned}
Jac_{-1} & \quad \{ x : \left( \frac{x}{N} \right) = -1 \} \\
Jac_{+1} & \quad \{ x : \left( \frac{x}{N} \right) = +1 \}
\end{aligned}
\]

Surprising fact: Jacobi symbol $\left( \frac{x}{N} \right) = \left( \frac{x}{p} \right) \left( \frac{x}{q} \right)$ is computable in poly time without knowing $p$ and $q$. 
Quadratic Residuosity

Let $N = pq$ be a product of two large primes.

So: $QR_N = \{x: \left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = +1\}$

$QNR_N = \{x: \left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = -1\}$

$QR_N$ is the set of squares mod $N$ and $QNR_N$ is the set of non-squares mod $N$ with Jacobi symbol +1.
Quadratic Residuosity

Exactly half residues even if

\[ N = p^i q^j, \ i, j \geq 1, \text{ not both even.} \]

\[ QR_N \]

\[ QNR_N \]

\( QR_N \) is the set of squares mod \( N \) and \( QNR_N \) is the set of non-squares mod \( N \) with Jacobi symbol +1.
Quadratic Residuosity

Exactly half residues even if
\[ N = p^i q^j, \quad i, j \geq 1, \text{not both even}. \]

IMPORTANT PROPERTY: If \( y_1 \) and \( y_2 \) are both in \( QNR \), then their product \( y_1 y_2 \) is in \( QR \).
The fraction of residues smaller if $N$ has three or more prime factors!

**IMPORTANT PROPERTY:** If $y_1$ and $y_2$ are both in $QNR$, then their product $y_1y_2$ is in $QR$. 
Quadratic Residuosity

Let $N = pq$ be a product of two large primes.

**Quadratic Residuosity Assumption (QRA)**

No PPT algorithm can distinguish between a random element of $QR_N$ from a random element of $QNR_N$ given only $N$. 
HOW TO CONSTRUCT NIZK IN THE CRS MODEL

Step 1. **Review** our number theory hammers & polish them.

Step 2. **Construct** NIZK for a special NP language, namely quadratic *non*-residuosity.

Step 3. **Bootstrap** to NIZK for 3SAT, an NP-complete language.
NIZK for Quadratic Non-Residuosity

Define the NP language $GOOD$ with instances $(N, y)$ where:

- $N$ is good: has exactly two prime factors and is not a perfect square; and

- $y \in QNR_N$ (that is, $y$ has Jacobi symbol +1 but is not a square mod $N$)
NIZK for Quadratic Non-Residuosity

$$CRS = (r_1, r_2, ..., r_m) \leftarrow (Jac_N^{+1})^m$$

If $N$ is good and $y \in QNR_N$:
- either $r_i$ is in $QR_N$ or $yr_i$ is in $QR_N$
- so I can compute $\sqrt{r_i}$ or $\sqrt{yr_i}$.

If not ... I’ll be stuck!
NIZK for Quadratic Non-Residuosity

\( CRS = (r_1, r_2, \ldots, r_m) \leftarrow (Jac_N^{+1})^m \)

\((N, y) \quad P \quad \forall i: \sqrt{r_i} \text{ OR } \sqrt{yr_i} \quad V \quad (N, y)\)

Check:
- \(N\) is not a prime power,
- \(N\) is not a perfect square; and
- I received either a mod-N square root of \(r_i\) or \(yr_i\).
NIZK for Quadratic Non-Residuosity

\[ CRS = (r_1, r_2, ..., r_m) \leftarrow (\text{Jac}_{N}^{+1})^m \]

Soundness (what if \( N \) has more than 2 prime factors)

No matter what \( y \) is, for half the \( r_i \), both \( r_i \) and \( yr_i \) are not quadratic residues.
NIZK for Quadratic Non-Residuosity

\[ CRS = (r_1, r_2, ..., r_m) \leftarrow (Jac_N^{+1})^m \]

Soundness (what if \(N\) has more than 2 prime factors)

No matter what \(y\) is, for half the \(r_i\), both \(r_i\) and \(yr_i\) are not quadratic residues.
NIZK for Quadratic Non-Residuosity

\[ CRS = (r_1, r_2, ..., r_m) \leftarrow (Jac_N^{+1})^m \]

\[(N, y) \quad \text{P} \quad \forall i: \sqrt{r_i} \text{ OR } \sqrt{yr_i} \quad \text{V} \quad (N, y)\]

**Soundness** (what if \( y \) is a residue)

Then, if \( r_i \) happens to be a non-residue, both \( r_i \) and \( yr_i \) are **not** quadratic residues.
NIZK for Quadratic Non-Residuosity

\[ CRS = (r_1, r_2, \ldots, r_m) \leftarrow (Jac_N^{+1})^m \]

\((N, y)\)

\[ P \quad \forall i: \pi_i = \sqrt{r_i} \text{ OR } \sqrt{yr_i} \quad \text{(Perfect) Zero Knowledge Simulator S:} \]

First pick the proof \(\pi_i\) to be random in \(Z_N^*\).

Then, reverse-engineer the CRS, letting \(r_i = \pi_i^2\) or \(r_i = \pi_i^2/y\) randomly.

\((N, y)\)
NIZK for Quadratic Non-Residuosity

$$CRS = (r_1, r_2, ..., r_m) \leftarrow (Jac_N^{+1})^m$$

CRS depends on the instance N. Not good.

**Soln:** Let CRS be random numbers. Interpret them as elements of $Z^*_N$ and both the prover and verifier filter out $Jac_N^{-1}$. 
Next Lecture

Step 1. **Review** our number theory hammers & polish them.

Step 2. **Construct** NIZK for a special NP language, namely quadratic *non*-residuosity.

Step 3. **Bootstrap** to NIZK for 3SAT, an NP-complete language.