MIT 6.875

Foundations of Cryptography
Lecture 16

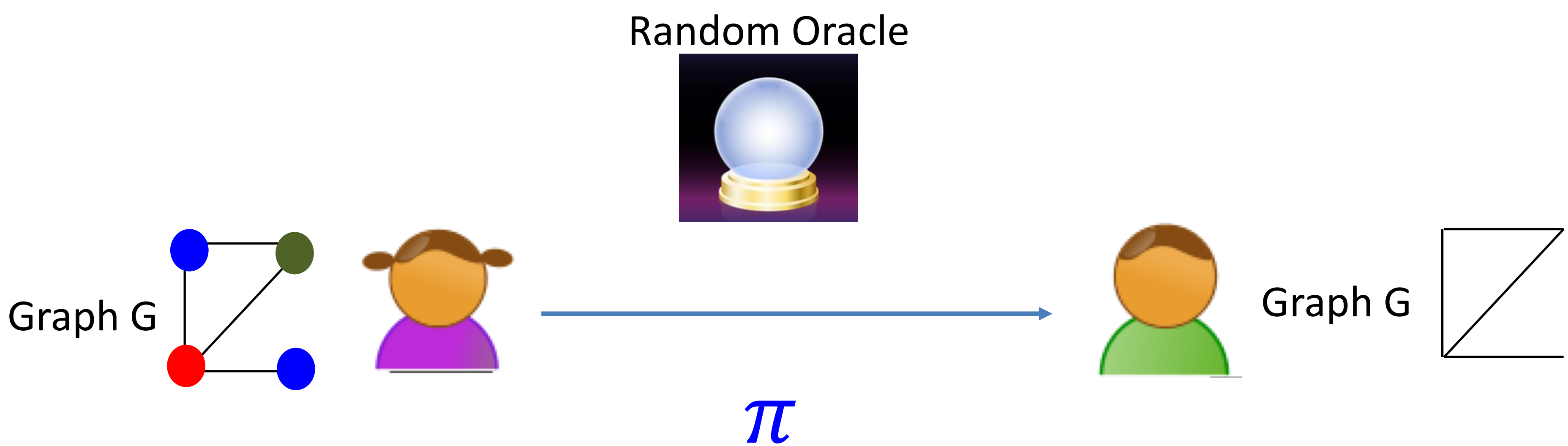
Interaction is Necessary for ZK

Theorem: If a language L has a non-interactive (one-message) ZK proof system, then L can be solved in probabilistic polynomial time.

That seems like the end of the road for non-interactive ZK (?)

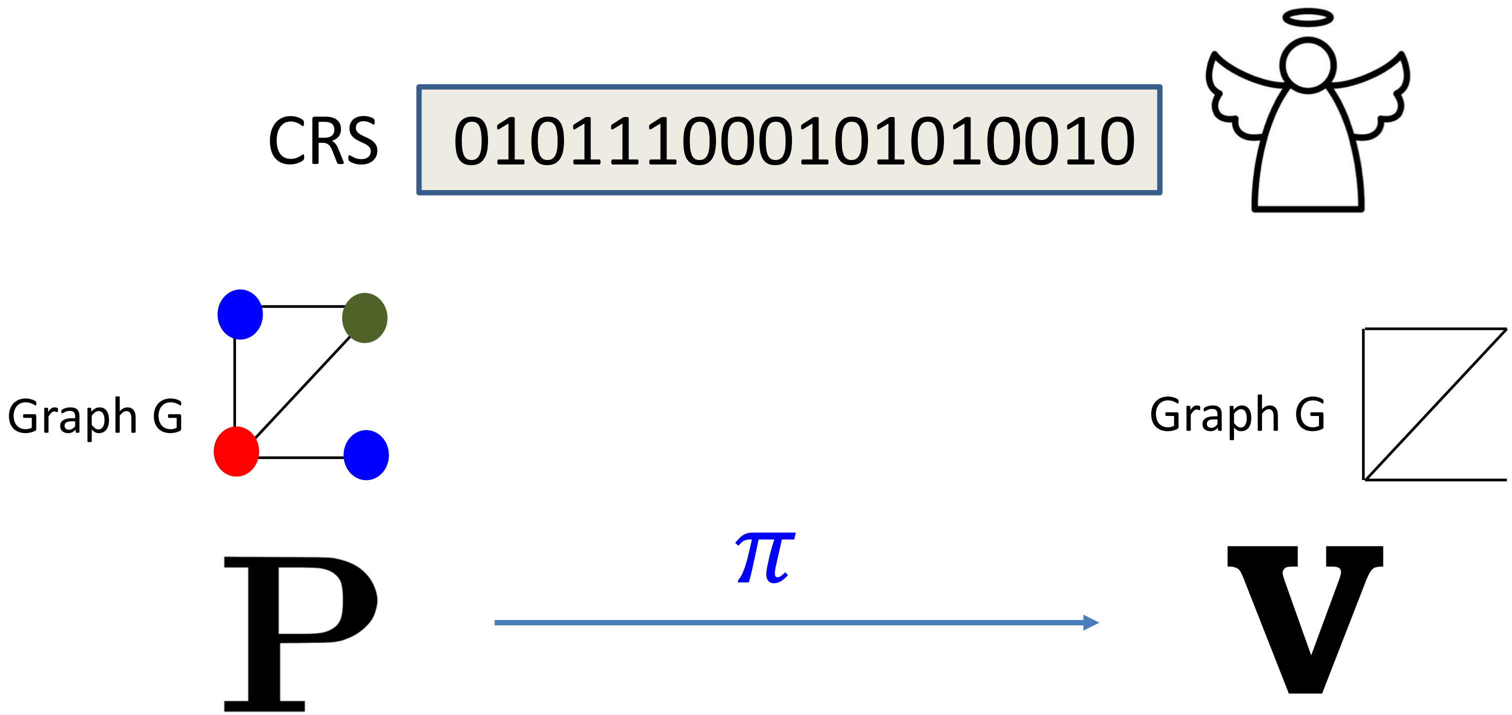
Two Roads to Non-Interactive ZK (NIZK)

1. Random Oracle Model & Fiat-Shamir Transform.

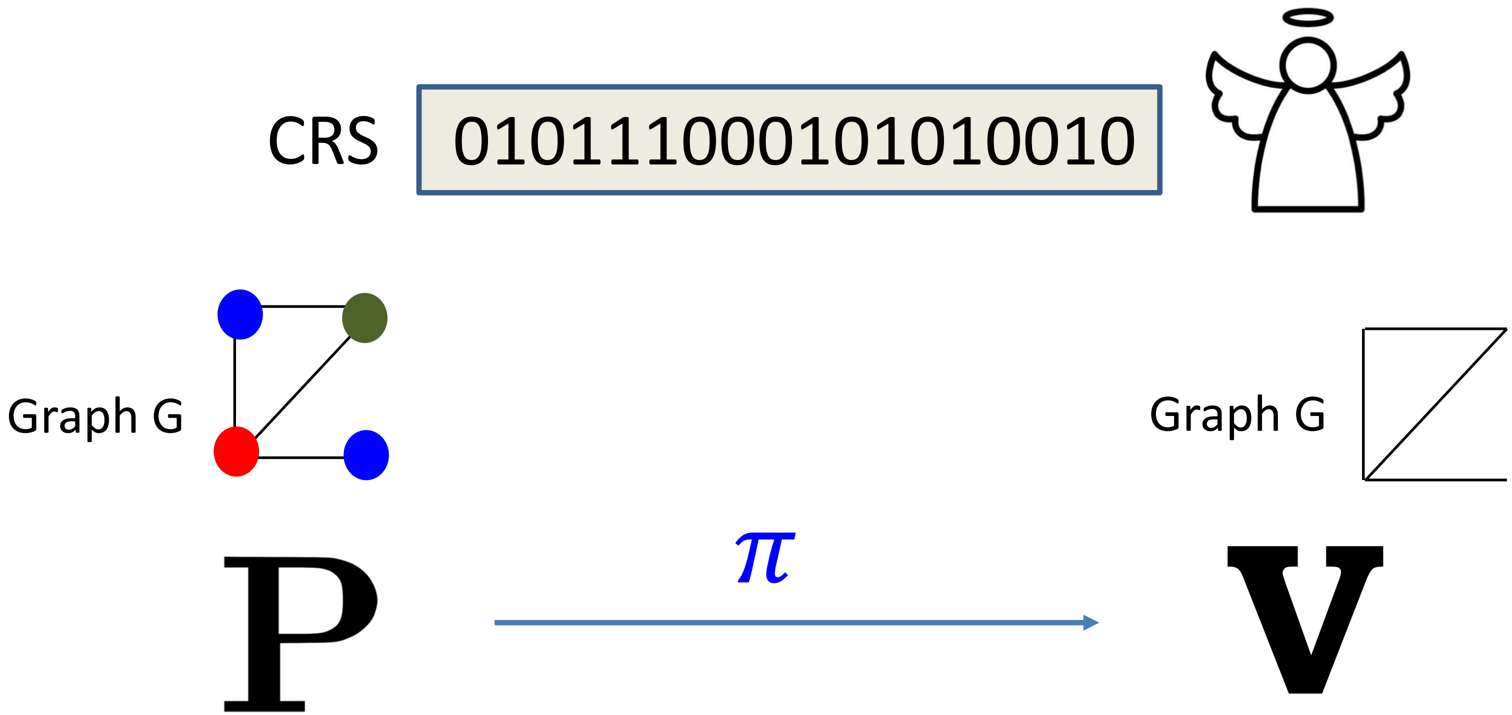


2. Common Random String Model.

The Common Random String Model



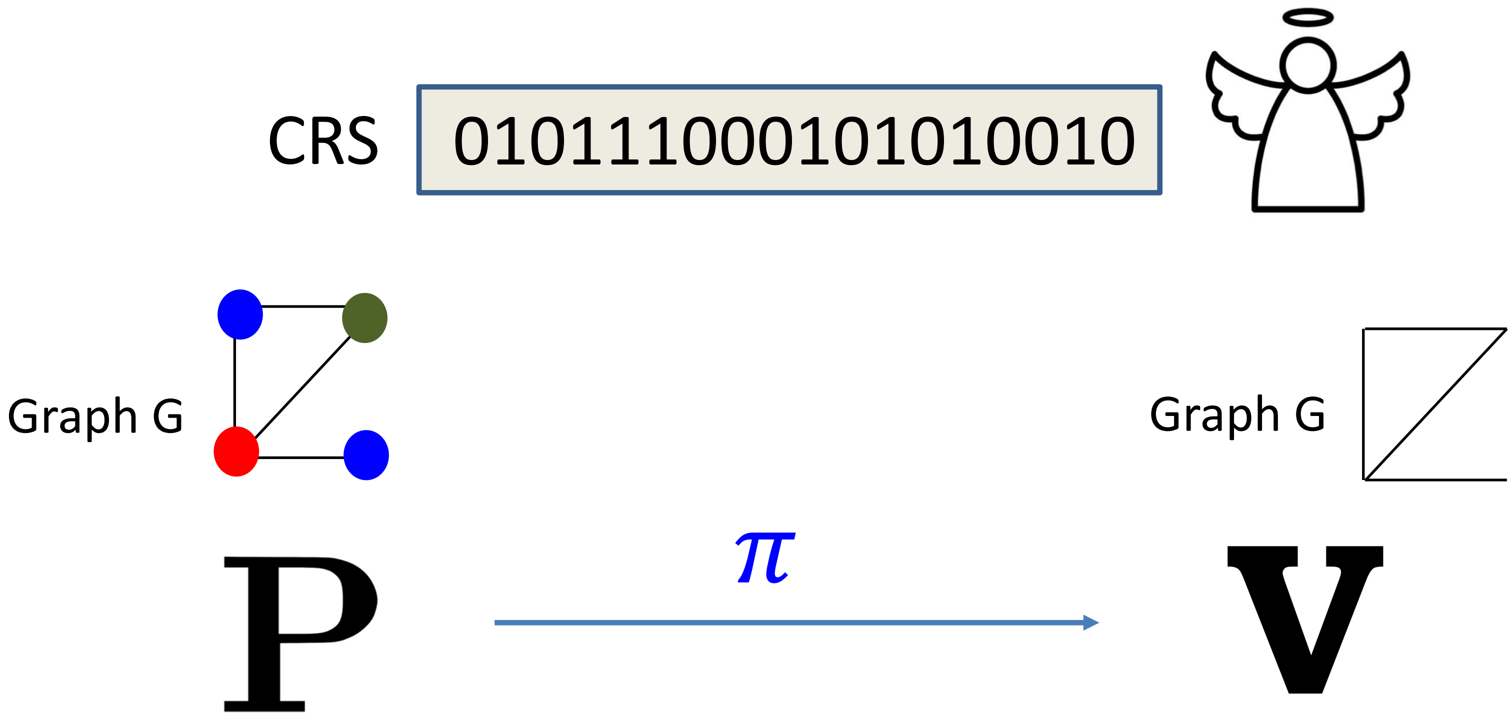
NIZK for a language L in the CRS Model



1. Completeness: For every $x \in$ L, V accepts P's proof.

2. Soundness: Given a CRS, the probability that a cheating prover P^* can produce $x^* \notin$ L and "proof" π^*, such that $V(CRS, x^*, \pi^*)$ accepts is $\leq \mathrm{neg}(n)$

NIZK for a language L in the CRS Model



3. Zero Knowledge: There is a PPT simulator S such that for every $x \in \mathrm{L}$ and witness w, S ***simulates the view*** of the verifier V.

$$S(x) \approx (CRS \leftarrow Unif, \pi \leftarrow P(CRS, x, w))$$

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-Rothblum^2-Wichs’19 and Peikert-Shiehian’19 *(learning with errors)*

HOW TO CONSTRUCT NIZK IN THE CRS MODEL

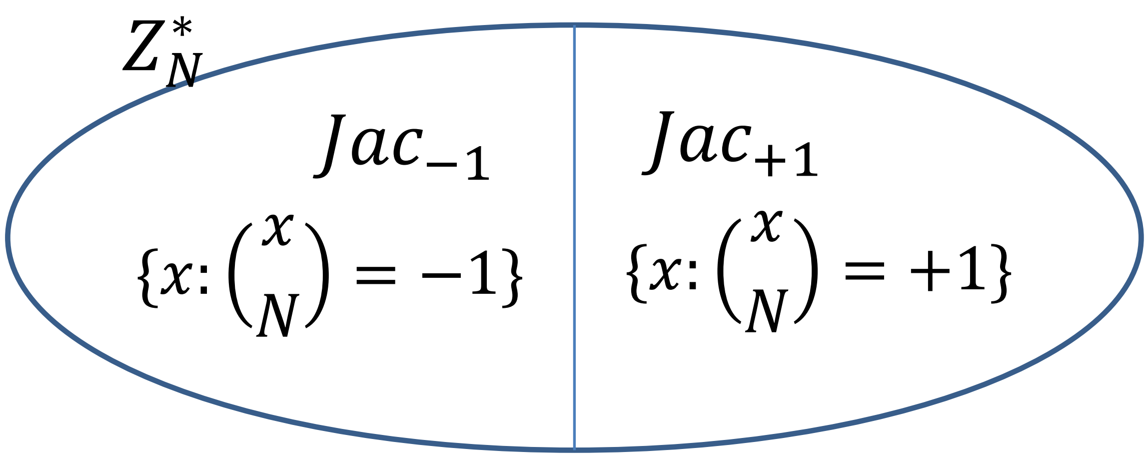
Step 1. **Review** our number theory hammers & polish them.

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

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

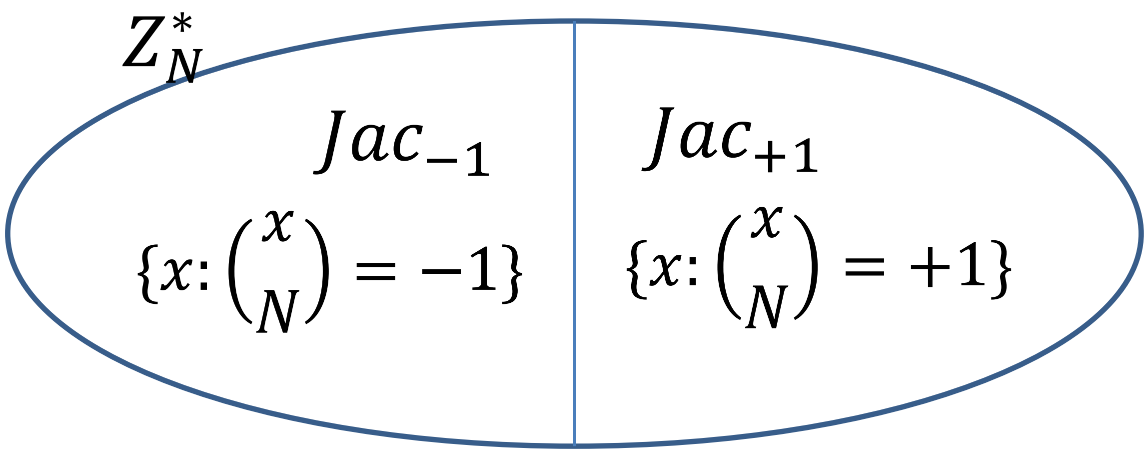
Jacobi Symbol

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



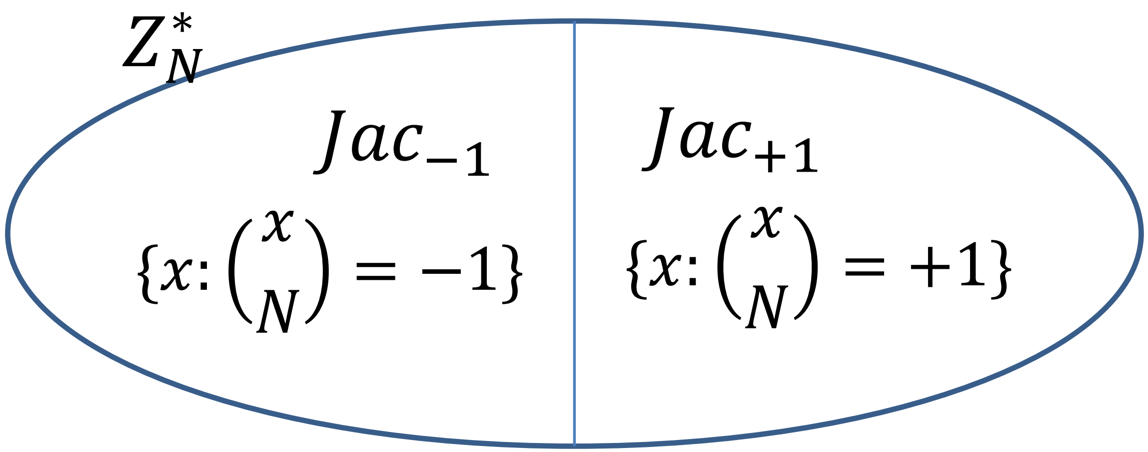
Jacobi Symbol

Fact: For any odd N, Jac divides Z_N^* evenly *unless N is a perfect square*. (If N is a perfect square, all of Z_N^* has Jacobi symbol +1.)



Jacobi Symbol

Surprising fact: For any N, Jacobi symbol $\left(\frac{x}{N}\right)$ is computable in poly time **without knowing the prime factorization of $\boldsymbol{N}$**.

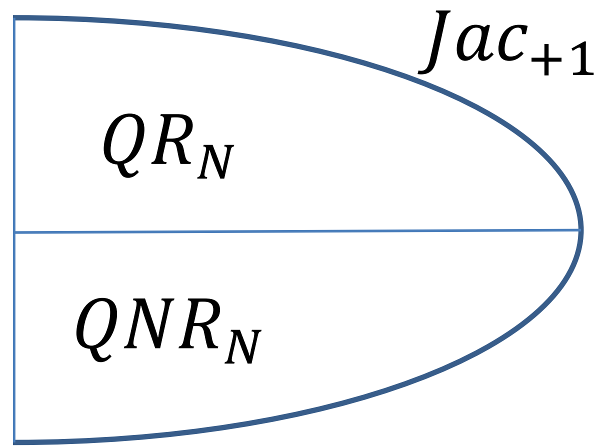


Quadratic Residues / Squares

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

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

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



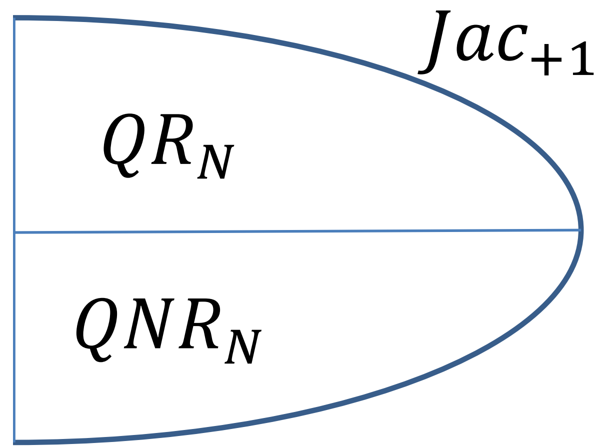
QR_N is the set of squares mod N and QNR_N is the set of non-squares mod N with Jacobi symbol +1.

Quadratic Residues / Squares

Fact: For an odd $N = \prod_{i=1}^{k} p_i^{\alpha_i}$, the fraction of Z_N^* that are a square mod N is 2^{-k}.

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

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

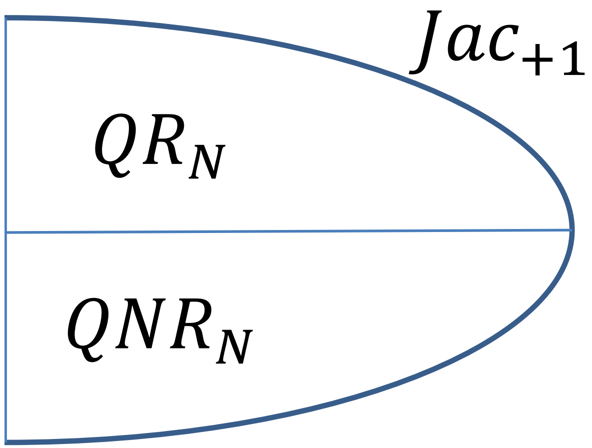


QR_N is the set of squares mod N and QNR_N is the set of non-squares mod N with Jacobi symbol +1.

Quadratic Residues

Call an odd integer N good if

- **exactly half of Z_N^* have Jacobi symbol +1, and**
- **exactly half of them are quadratic residues.**

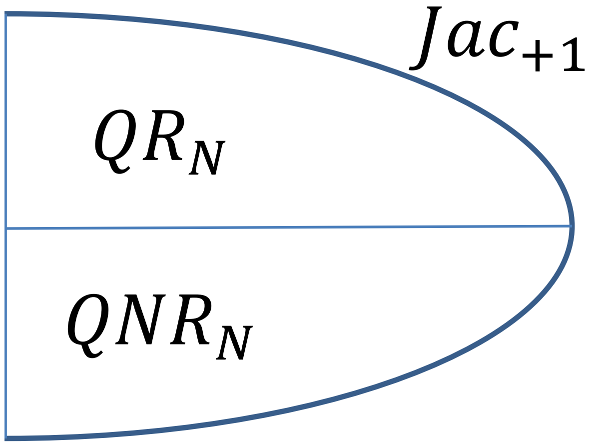


QR_N is the set of squares mod N and QNR_N is the set of non-squares mod N with Jacobi symbol +1.

Quadratic Residues

Fact: An odd N is good iff

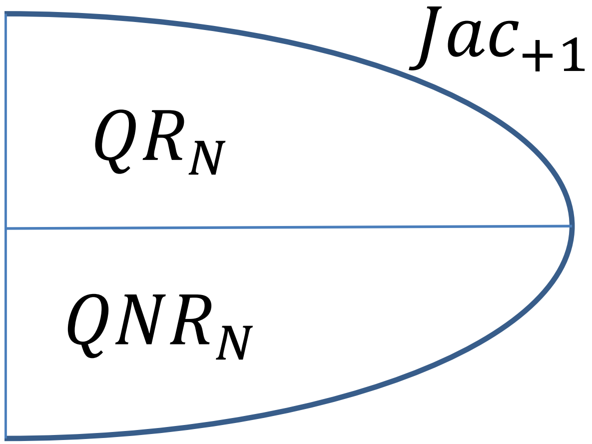
$$N = p^i q^j, and\ i, j \geq 1, \text{not both even.}$$



Quadratic Residues

Fact: An odd N is good iff

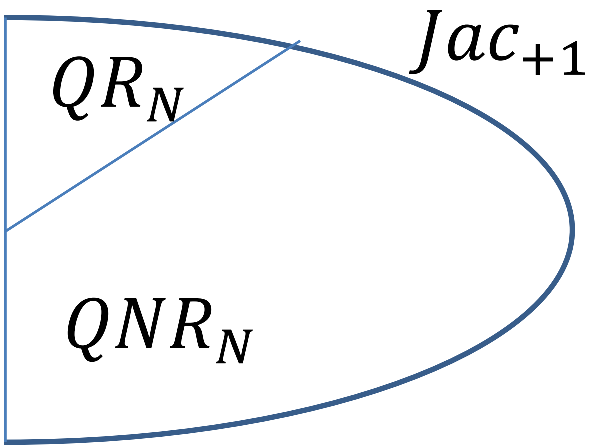
$$N = p^i q^j, and\ i, j \geq 1, \text{not both even.}$$



IMPORTANT PROPERTY: If y_1 and y_2 are both in $Q\boldsymbol{N}R$, then their product $y_1 y_2$ is in QR.

Quadratic Residues

The fraction of residues smaller if N has three or more prime factors!



~~**<u>IMPORTANT PROPERTY</u>**: If y_1 and y_2 are both in $Q\boldsymbol{N}R$, then their product $y_1 y_2$ is in QR.~~

Quadratic Residuosity

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

<u>Quadratic Residuosity Assumption (QRA)</u>

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.

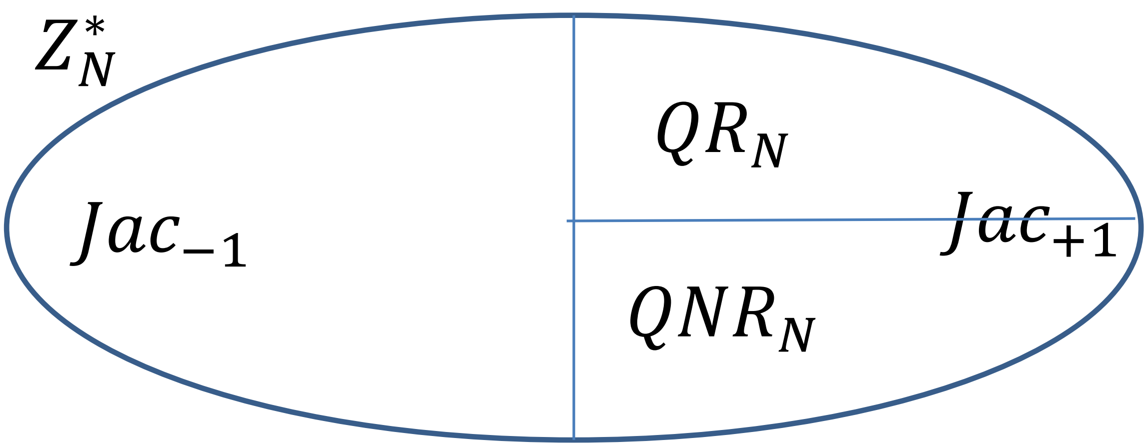
Step 2. **Construct** NIZK for a special NP language, namely quadratic *non*-residuosity.

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

NIZK for Quadratic Non-Residuosity

Define the NP language $GOOD$ with instances $(\boldsymbol{N}, \boldsymbol{y})$ where

- N is good; and
- $y \in QNR_N$ (that is, y has Jacobi symbol +1 but is not a square mod N)



NIZK for Quadratic Non-Residuosity

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

(N, y) (N, y)

$\mathbf{P} \longrightarrow \mathbf{V}$

Check:

- N is odd
- N is not a prime power,
- N is not a perfect square;

Fact: If all these pass, then at most half of Jac_N^{+1} are squares.

NIZK for Quadratic Non-Residuosity

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

(N, y) (N, y)

$$\mathbf{P} \longrightarrow \mathbf{V}$$

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

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

(N, y) **P**

$\forall i$: $\sqrt{r_i}$ OR $\sqrt{yr_i}$ $\longrightarrow$

(N, y) **V**

Check:

- N is odd
- 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

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

(N, y) $\qquad\qquad\qquad\qquad$ (N, y)

$$\mathbf{P} \xrightarrow{\forall i:\ \sqrt{r_i} \text{ OR } \sqrt{yr_i}} \mathbf{V}$$

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

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

(N, y) (N, y)

P $\xrightarrow{\forall i:\ \sqrt{r_i}\ \text{OR}\ \sqrt{yr_i}}$ **V**

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

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

(N, y) **P** $\xrightarrow{\forall i:\ \sqrt{r_i}\ \text{OR}\ \sqrt{y r_i}}$ **V** (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

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

(N, y) (N, y)

$$\mathbf{P} \xrightarrow{\forall i: \pi_i = \sqrt{r_i} \text{ OR } \sqrt{y r_i}} \mathbf{V}$$

(Perfect) Zero Knowledge Simulator S:

First pick the proof π_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.

$\blacksquare$

NIZK for Quadratic Non-Residuosity

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

(N, y) $\qquad\qquad\qquad\qquad$ (N, y)

$$\mathbf{P} \longrightarrow \mathbf{V}$$

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.

3SAT

Boolean Variables: x_i can be either true (1) or false (0)

A Literal is either x_i or $\overline{x_i}$.

A Clause is a *disjunction* of literals.

E.g. $x_1 \vee x_2 \vee \overline{x_5}$

A Clause is true if any one of the literals is true.

3SAT

Boolean Variables: x_i can be either true (1) or false (0)

A Literal is either x_i or $\overline{x_i}$.

A Clause is a *disjunction* of literals.

E.g. $x_1 \vee x_2 \vee \overline{x_5}$ is true as long as:

$$(x_1, x_2, x_5) \neq (0,0,1)$$

3SAT

Boolean Variables: x_i can be either true (1) or false (0)

A Literal is either x_i or $\overline{x_i}$.

A 3-Clause is a *disjunction* of 3-literals.

A 3-SAT formula is a *conjunction* of many 3-clauses.

E.g. $\boldsymbol{\Psi} = (x_1 \vee x_2 \vee \overline{x_5}) \wedge (x_1 \vee x_3 \vee x_4)\ (\overline{x_2} \vee x_3 \vee \overline{x_5})$

A 3-SAT formula $\boldsymbol{\Psi}$ is **satisfiable** if there is an assignment of values to the variables x_i that makes all its clauses true.

3SAT

Cook-Levin Theorem: It is NP-complete to decide whether a <u>3-SAT formula</u> $\mathbf{\Psi}$ is satisfiable.

A <u>3-SAT formula</u> is a *conjunction* of many 3-clauses.

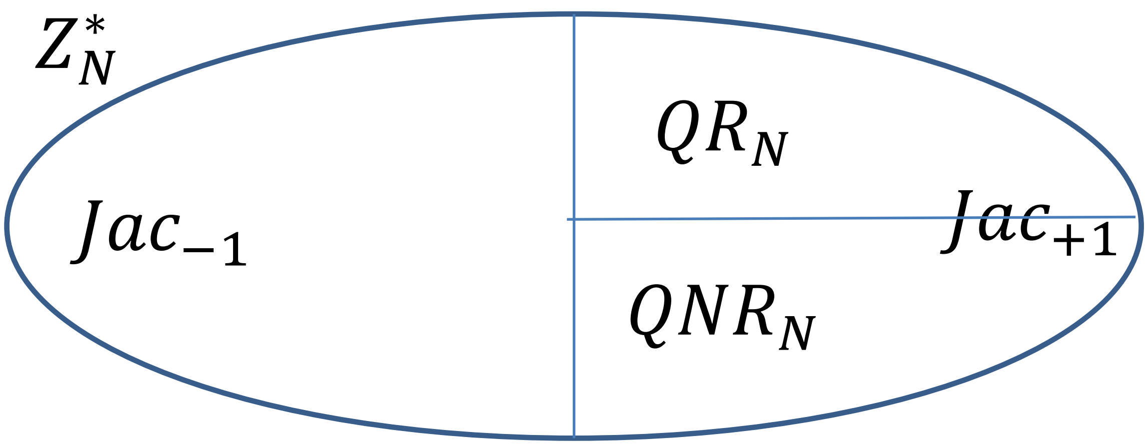
E.g. $\mathbf{\Psi} = (x_1 \vee x_2 \vee \overline{x_5}) \wedge (x_1 \vee x_3 \vee x_4)\ (\overline{x_2} \vee x_3 \vee \overline{x_5})$

A <u>3-SAT formula</u> $\mathbf{\Psi}$ is **satisfiable** if there is an assignment of values to the variables x_i that makes all its clauses true.

NIZK for 3SAT: Recall…

We saw a way to show that a pair $(\boldsymbol{N}, \boldsymbol{y})$ is GOOD. That is:

- the following is the picture of Z_N^* and
- for every $r \in Jac_{+1}$, either r or ry is a quadratic residue.



NIZK for 3SAT

$$CRS = (r_1, r_2, \ldots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

$\mathbf{\Psi}$ $\mathbf{\Psi}$

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$ **P** $\xrightarrow{(N, y, \pi)}$ **V**

1. Prover picks an (N, y) and proves that it is GOOD.

Input: $\mathbf{\Psi}$ $= (x_1 \vee x_2 \vee \overline{x_5}) \wedge (x_1 \vee x_3 \vee x_4)\ (\overline{x_2} \vee x_3 \vee \overline{x_5})$

n variables, m clauses.

NIZK for 3SAT

$$CRS = (r_1, r_2, \ldots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

Ψ Ψ

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$ **P** $\xrightarrow{(N, y, \pi)}$ **V**

2. Prover encodes the satisfying assignment

$y_i \leftarrow QR_N$ if x_i is false

$y_i \leftarrow QNR_N$ if x_i is true

NIZK for 3SAT

$$CRS = (r_1, r_2, \ldots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

$\boldsymbol{\Psi}$ $\boldsymbol{\Psi}$

Satisfying assignment
$(w_1, w_2, \ldots, w_n)$ $\mathbf{P} \xrightarrow{(N, y, \pi)} \mathbf{V}$

2. Prover encodes the satisfying assignment & ∴ the literals

$Enc(x_i) = y_i$, then $Enc(\overline{x_i}) = yy_i$

∴ exactly one of $Enc(x_i)$ or $Enc(\overline{x_i})$ is a non-residue.

NIZK for 3SAT

$$CRS = (r_1, r_2, \dots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

$\boldsymbol{\Psi}$ (N, y, π) $\boldsymbol{\Psi}$

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_n)$ **P** Encode vars: $(y_1, \dots, y_n)$ **V**

2. Prover encodes the satisfying assignment & ∴ the literals

$Enc(x_i) = y_i$, then $Enc(\overline{x_i}) = yy_i$

∴ exactly one of $Enc(x_i)$ or $Enc(\overline{x_i})$ is a non-residue.

NIZK for 3SAT

$$CRS = (r_1, r_2, \ldots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

Ψ $\qquad\qquad (N, y, \pi) \qquad\qquad$ Ψ

Satisfying assignment $(w_1, w_2, \ldots, w_n)$ **P** $\xrightarrow{\text{Encode vars: } (y_1, \ldots, y_n)}$ **V**

3. Prove that (encoded) assignment satisfies each clause.

For each clause, say $x_1 \vee x_2 \vee \overline{x_5}$, let $(a_1 = y_1, b_1 = y_2, c_1 = yy_5)$

For each clause, say $x_1 \vee x_2 \vee \overline{x_5}$, let (a_1, b_1, c_1) denote the encoded variables.

So, each of them is either y_i (if the literal is a var) or yy_i (if the literal is a negated var).

NIZK for 3SAT

$$CRS = (r_1, r_2, \dots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

Ψ (N, y, π) Ψ

Satisfying assignment $(w_1, w_2, \dots, w_n)$ **P** Encode vars: $(y_1, \dots, y_n)$ $\rightarrow$ **V**

3. Prove that (encoded) assignment satisfies each clause.

For each clause, say $x_1 \vee x_2 \vee \overline{x_5}$,
let (a_1, b_1, c_1) denote the encoded variables.

WANT to SHOW: $x_1\ OR\ x_2\ OR\ \overline{x_5}$ is true.

NIZK for 3SAT

$$CRS = (r_1, r_2, \ldots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

Ψ (N, y, π) Ψ

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$ **P** Encode vars: $(y_1, \ldots, y_n)$ → **V**

3. Prove that (encoded) assignment satisfies each

For each clause, say $x_1 \vee x_2 \vee \overline{x_5}$,

let (a_1, b_1, c_1) denote the encoded variables.

WANT to SHOW: $a_1\ OR\ b_1\ OR\ c_1$ is a non-residue.

NIZK for 3SAT

Prove that (encoded) assignment satisfies each clause.

WANT to SHOW: $a_1 \ OR \ b_1 \ OR \ c_1$ is a non-residue.

Equiv: The "pattern" of (a_1, b_1, c_1) is **NOT** (QR, QR, QR).

CLEVER IDEA: Generate seven *additional* triples

original triple

show this is a QR:
reveal the square roots

(a_1, b_1, c_1)
(a_2, b_2, c_2)
(a_3, b_3, c_3)
(a_4, b_4, c_4)
(a_5, b_5, c_5)
(a_6, b_6, c_6)
(a_7, b_7, c_7)
(a_8, b_8, c_8)

"Proof of Coverage":
show that the 8 triples span
all possible QR patterns

NIZK for 3SAT

CLEVER IDEA: Generate seven *additional* triples

original triple

show this is a QR:
reveal the square roots

$$\left.\begin{matrix}(a_1, b_1, c_1)\\(a_2, b_2, c_2)\\(a_3, b_3, c_3)\\(a_4, b_4, c_4)\\(a_5, b_5, c_5)\\(a_6, b_6, c_6)\\(a_7, b_7, c_7)\\(a_8, b_8, c_8)\end{matrix}\right\}$$

"Proof of Coverage":
show that the 8 triples span all possible QR signatures

<u>**Proof of Coverage**</u>**:** For each of poly many triples (r, s, t) from CRS, show one of the 8 triples has the same signature.

That is, there is a triple (a_i, b_i, c_i) s.t. (ra_i, sb_i, tc_i) is (QR, QR, QR).

NIZK for 3SAT

$$CRS = (r_1, r_2, \ldots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

$\boldsymbol{\Psi}$ (N, y, π) $\boldsymbol{\Psi}$

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$ **P** $\xrightarrow{\text{Encode vars: } (y_1, \ldots, y_n)}$ **V**

For each clause ψ: ρ_ψ

3. Prove that (encoded) assignment satisfies each clause.

For each clause, construct the proof ρ = (7 additional triples, square root of the second triples, proof of coverage).

NIZK for 3SAT

$$CRS = (r_1, r_2, \dots, r_{large\ number}) \leftarrow (Jac_N^{+1})^{large\ number}$$

$\mathbf{\Psi}$ (N, y, π) $\mathbf{\Psi}$

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_\mathrm{n})$ **P** $\xrightarrow{\text{Encode vars: } (y_1, \dots, y_n)}$ **V**

For each clause ψ: ρ_ψ

Completeness & Soundness: Exercise.

Zero Knowledge: Simulator picks (N, y) where y is a quadratic **residue**.

Now, encodings of ALL the literals can be set to TRUE!!