MIT 6.875 & Berkeley CS276

Foundations of Cryptography
Lecture 17

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.

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. $x_1 \lor x_2 \lor \overline{x_5}$

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. $x_1 \lor x_2 \lor \overline{x_5}$ 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 3-SAT formula $\mathbf{\Psi}$ is satisfiable.

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

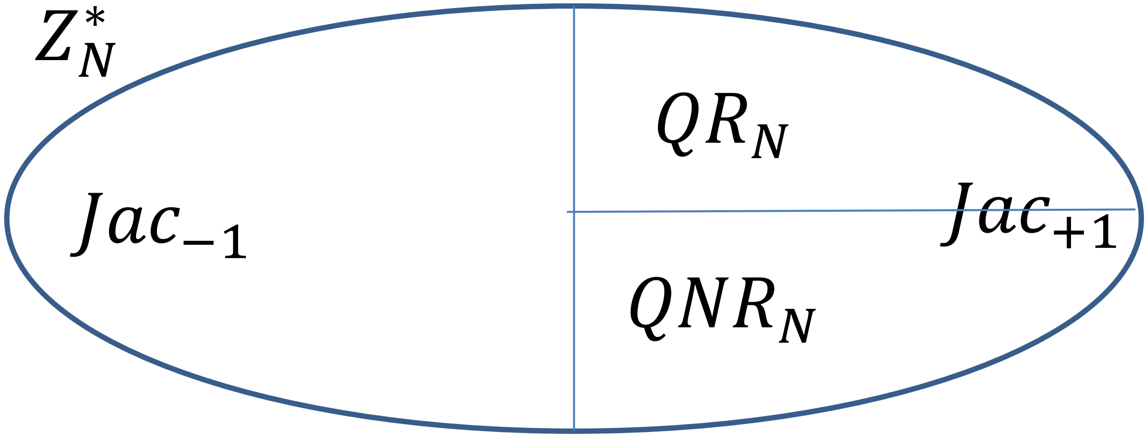
E.g. $\mathbf{\Psi} = (x_1 \vee x_2 \vee \overline{x_5}) \wedge (x_1 \vee x_3 \vee x_4)\ (\overline{x_2} \vee x_3 \vee \overline{x_5})$

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

NIZK for 3SAT: Recall…

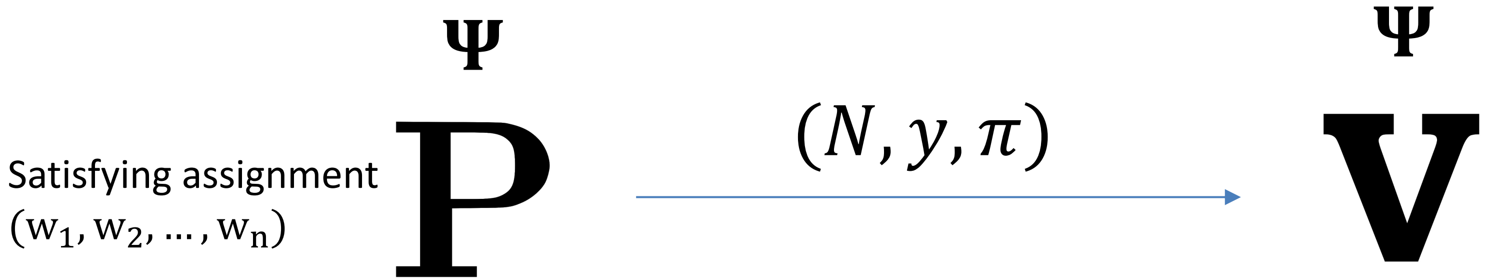
We saw a way to show that a pair $(\boldsymbol{N}, \boldsymbol{y})$ is GOOD. That is:

- 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

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



Ψ

Satisfying assignment
$(w_1, w_2, \dots, w_n)$

P

(N, y, π)

Ψ

V

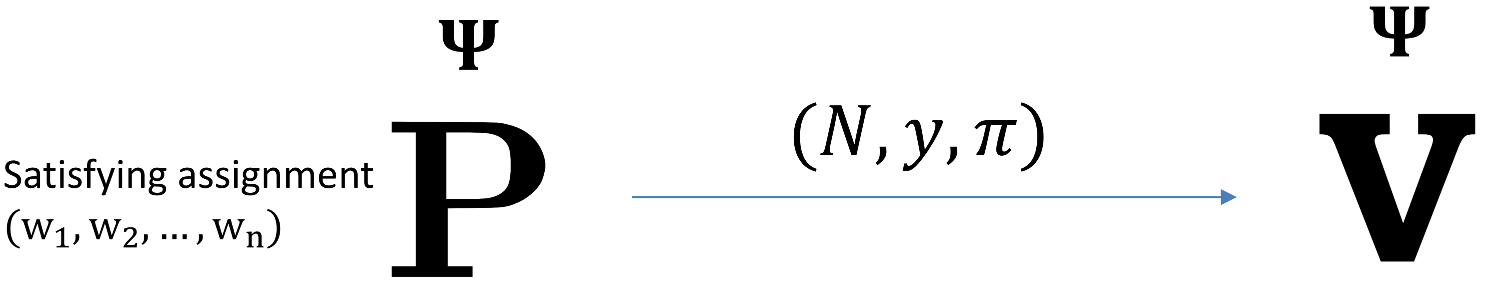
1. Prover picks an (N, y) and proves that it is GOOD.

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

n variables, m clauses.

NIZK for 3SAT

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



Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$

$\boldsymbol{\Psi}$

$\mathbf{P}$

(N, y, π)

$\boldsymbol{\Psi}$

$\mathbf{V}$

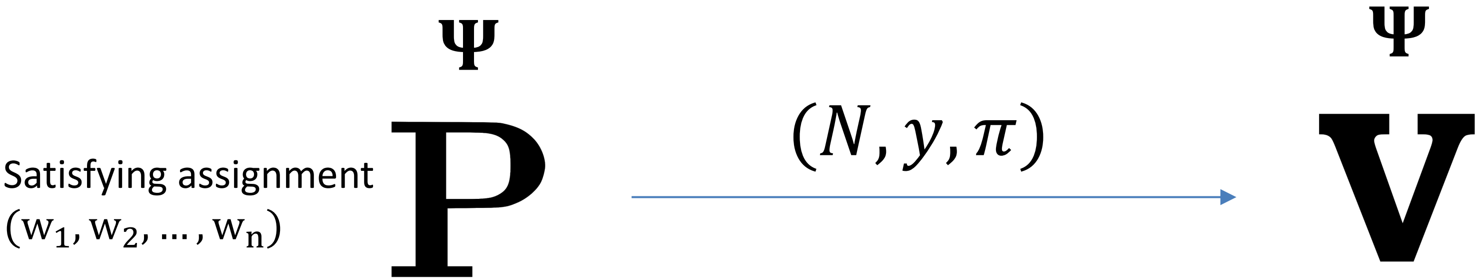
2. Prover encodes the satisfying assignment

$y_i \leftarrow QR_N$ if x_i is false

$y_i \leftarrow QNR_N$ if x_i is true

NIZK for 3SAT

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



Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_\mathrm{n})$

$\boldsymbol{\Psi}$

$\mathbf{P}$

(N, y, π)

$\boldsymbol{\Psi}$

$\mathbf{V}$

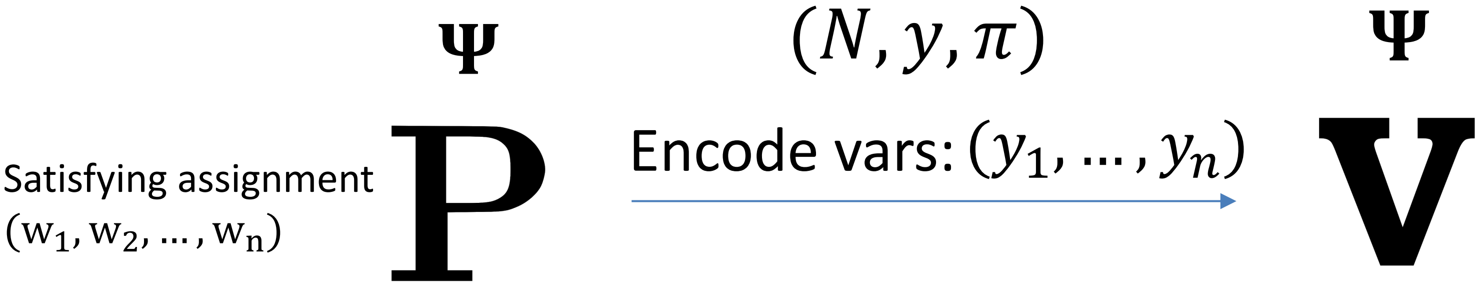
2. Prover encodes the satisfying assignment & ∴ the literals

$Enc(x_i) = y_i$, 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

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



$\boldsymbol{\Psi}$

(N, y, π)

$\boldsymbol{\Psi}$

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$

$\mathbf{P}$

Encode vars: $(y_1, \ldots, y_n)$

$\mathbf{V}$

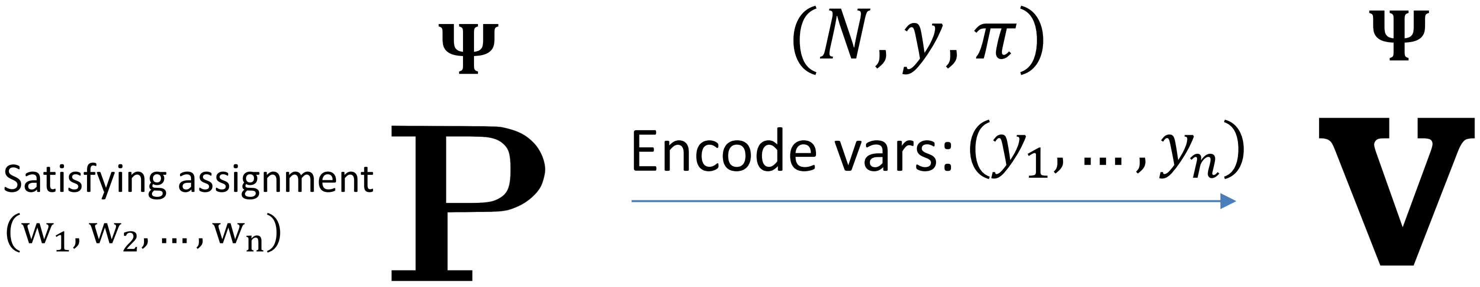
2. Prover encodes the satisfying assignment & ∴ the literals

$Enc(x_i) = y_i$, 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

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



Ψ (N, y, π) Ψ

Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_\mathrm{n})$ **P**

Encode vars: $(y_1, \dots, y_n)$

V

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

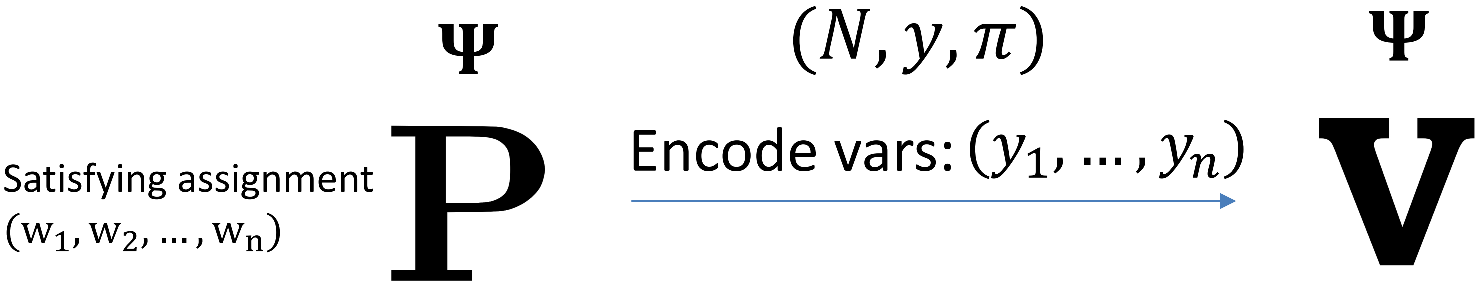
For each clause, say $x_1 \vee x_2 \vee \overline{x_5}$, let $(a_1 = y_1, b_1 = y_2, c_1 = yy_5)$ denote the encoded variables.

For each clause, say $x_1 \vee x_2 \vee \overline{x_3}$, 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 $(\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_n)$

$\mathbf{P}$

Encode vars: $(y_1, \dots, y_n)$

$\mathbf{V}$

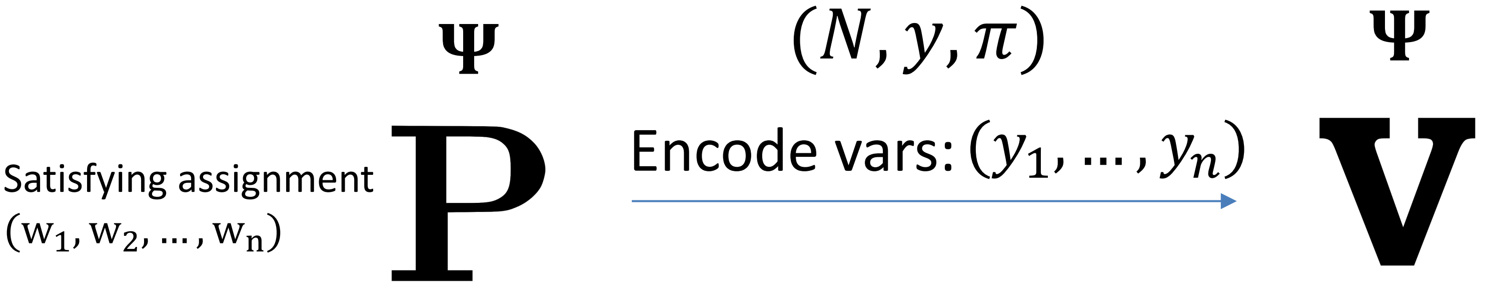
3. Prove that (encoded) assignment satisfies each clause.

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

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



$\boldsymbol{\Psi}$

(N, y, π)

$\boldsymbol{\Psi}$

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 V x_2 V $\overline{x_5}$,
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.

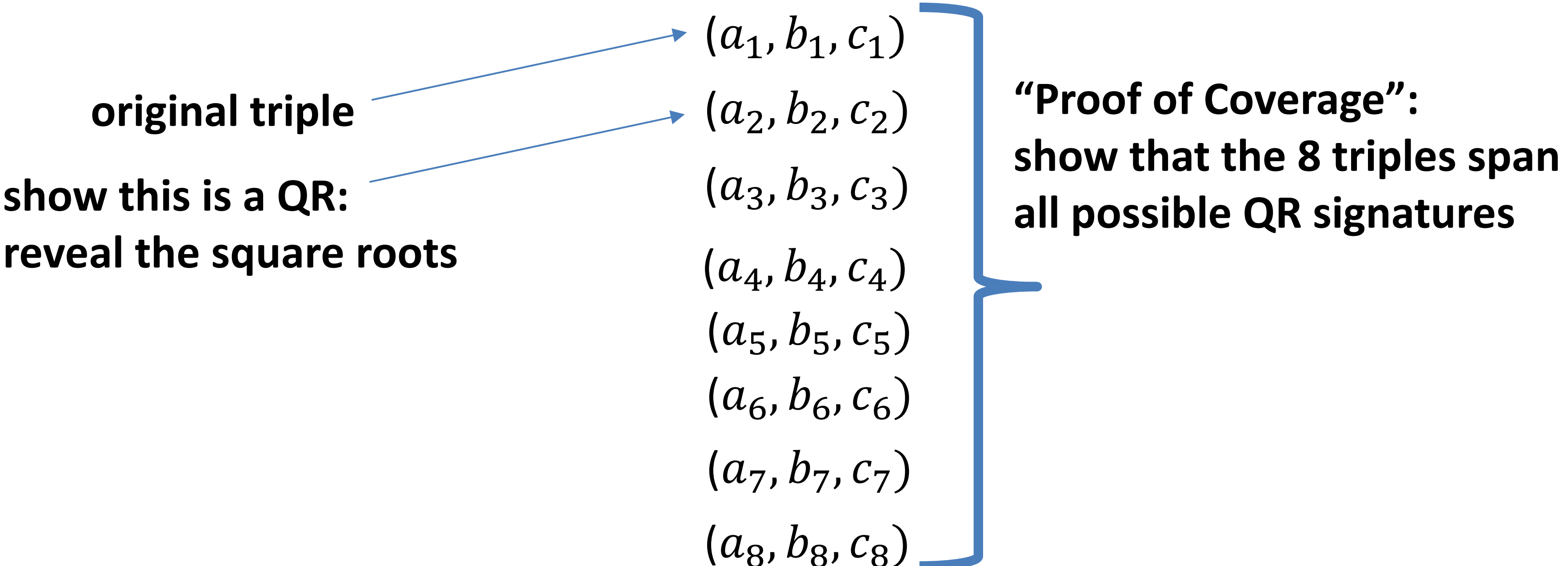
NIZK for 3SAT

Prove that (encoded) assignment satisfies each clause.

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

Equiv: The “signature” of (a_1, b_1, c_1) is **NOT** (QR, QR, QR).

CLEVER IDEA: Generate seven *additional* triples



(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)

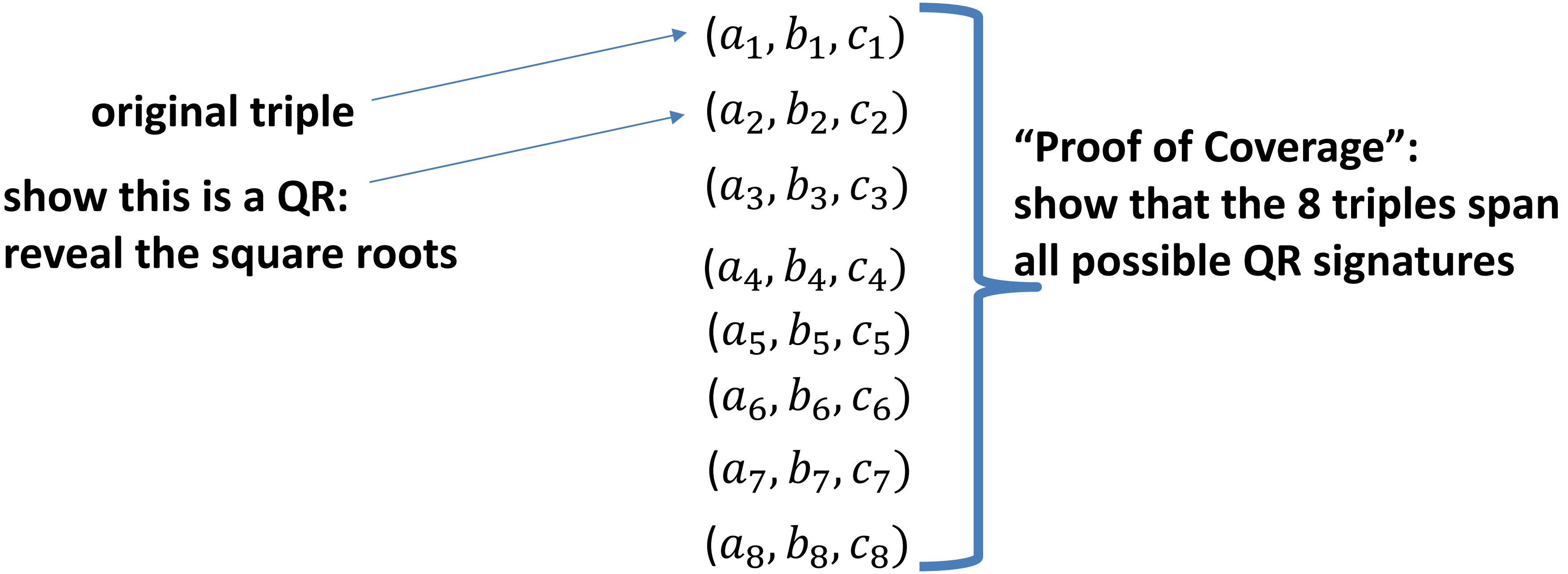
original triple

show this is a QR:
reveal the square roots

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

NIZK for 3SAT

CLEVER IDEA: Generate seven *additional* triples



(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)

original triple

show this is a QR:
reveal the square roots

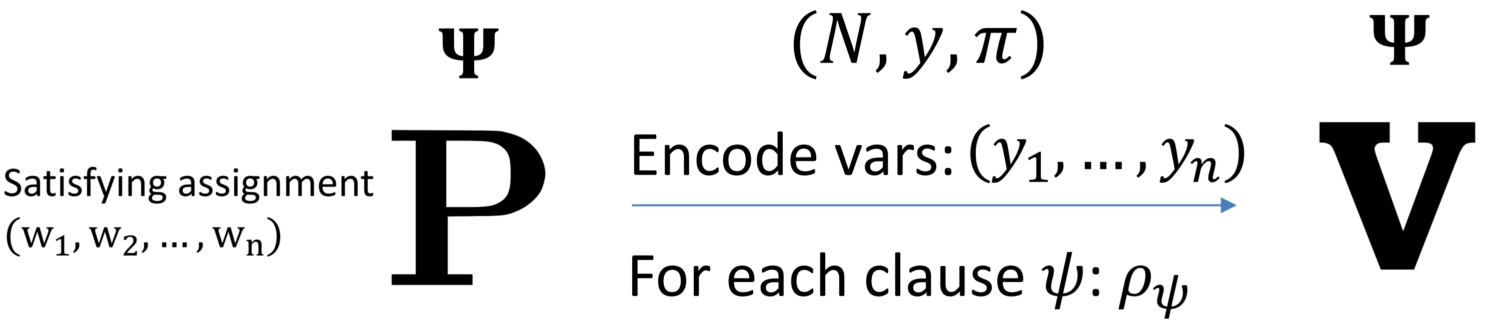
"Proof of Coverage":
show that the 8 triples span
all possible QR signatures

<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

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



Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$

$\boldsymbol{\Psi}$ **P**

(N, y, π)

Encode vars: $(y_1, \ldots, y_n)$

For each clause ψ: ρ_ψ

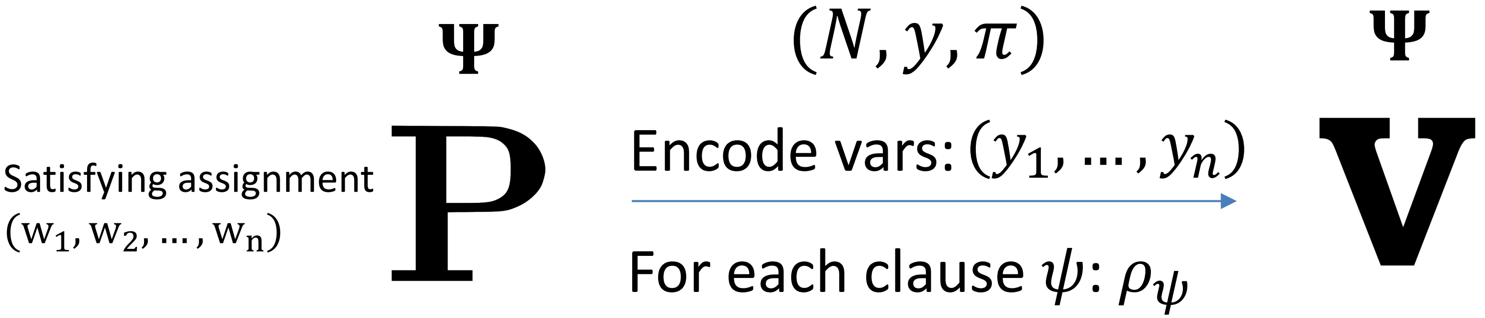
$\boldsymbol{\Psi}$ **V**

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

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



Satisfying assignment $(\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_\mathrm{n})$

$\boldsymbol{\Psi}$ $\mathbf{P}$

(N, y, π)

Encode vars: $(y_1, \dots, y_n)$

For each clause ψ: ρ_ψ

$\boldsymbol{\Psi}$ $\mathbf{V}$

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!!

HOW TO CONSTRUCT NIZK IN THE CRS MODEL

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

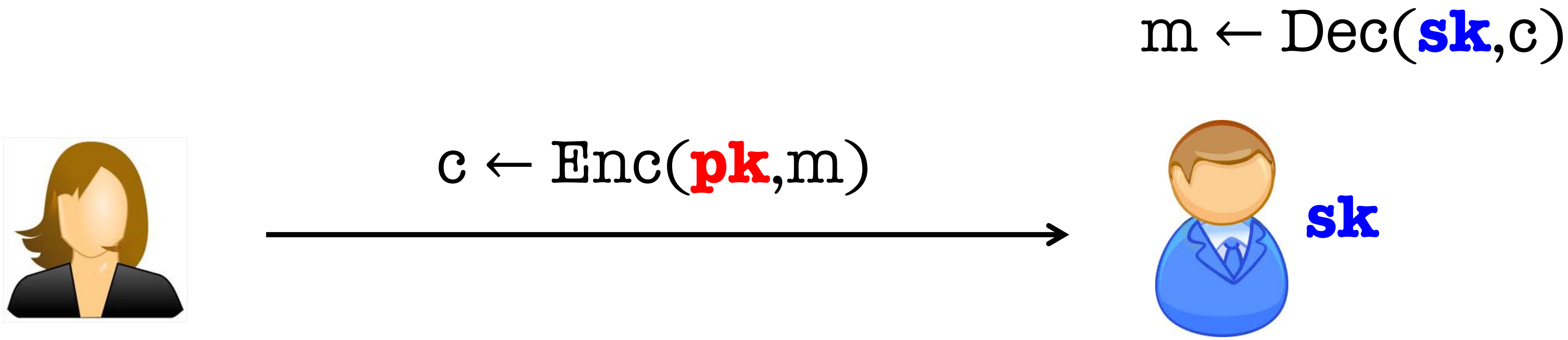
Step 2. **Construct** NIZK for a special NP language, namely
quadratic *non*-residuosity.

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

An Application of NIZK:

Non-malleable and Chosen Ciphertext Secure Encryption Schemes

Non-Malleability



$m \leftarrow \text{Dec}(\mathbf{sk}, c)$

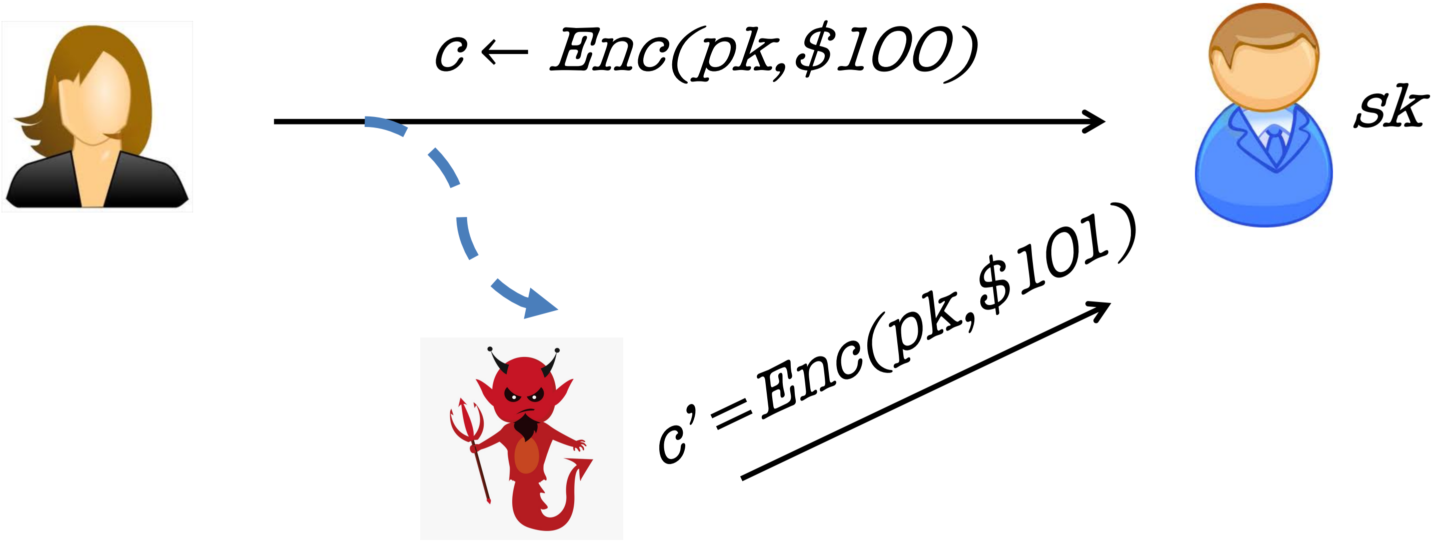
$c \leftarrow \text{Enc}(\mathbf{pk}, m)$

sk

Public-key directory

Bob	**pk**

Active Attacks 1: Malleability



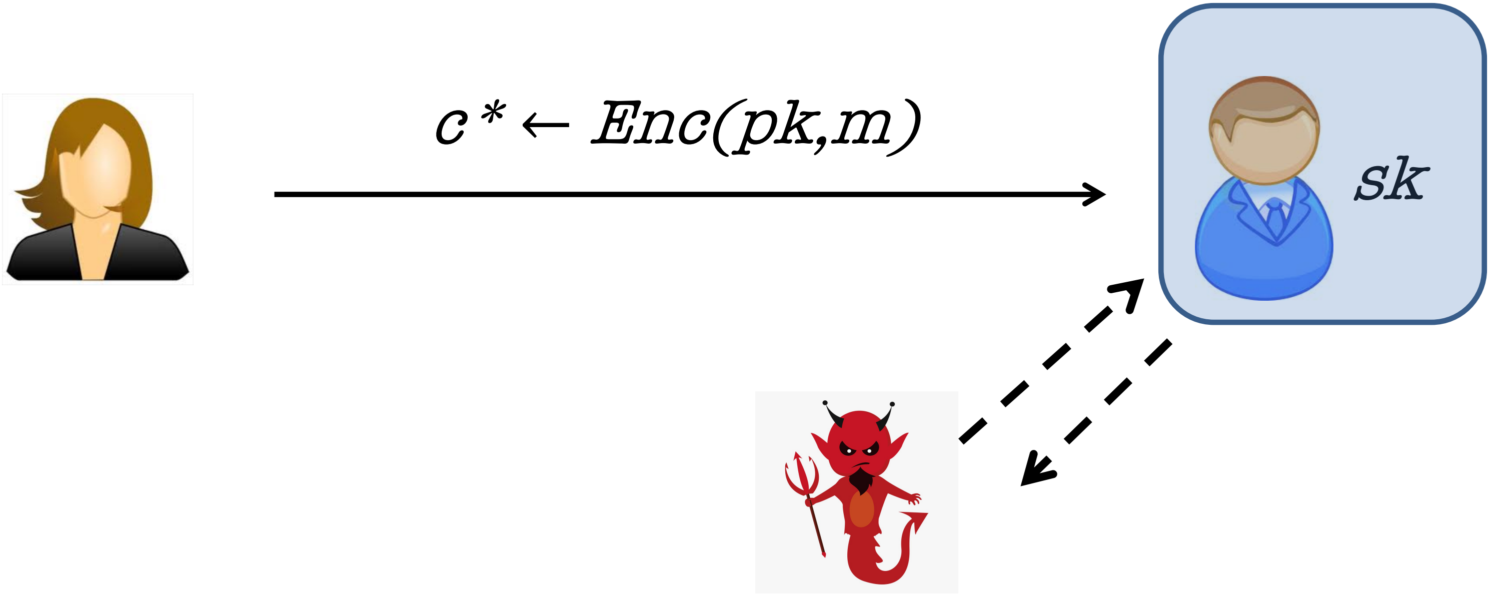
$c \leftarrow Enc(pk, \$100)$

sk

$c' = Enc(pk, \$101)$

ATTACK: Adversary could modify ("maul") an encryption of m into an encryption of a related message m'.

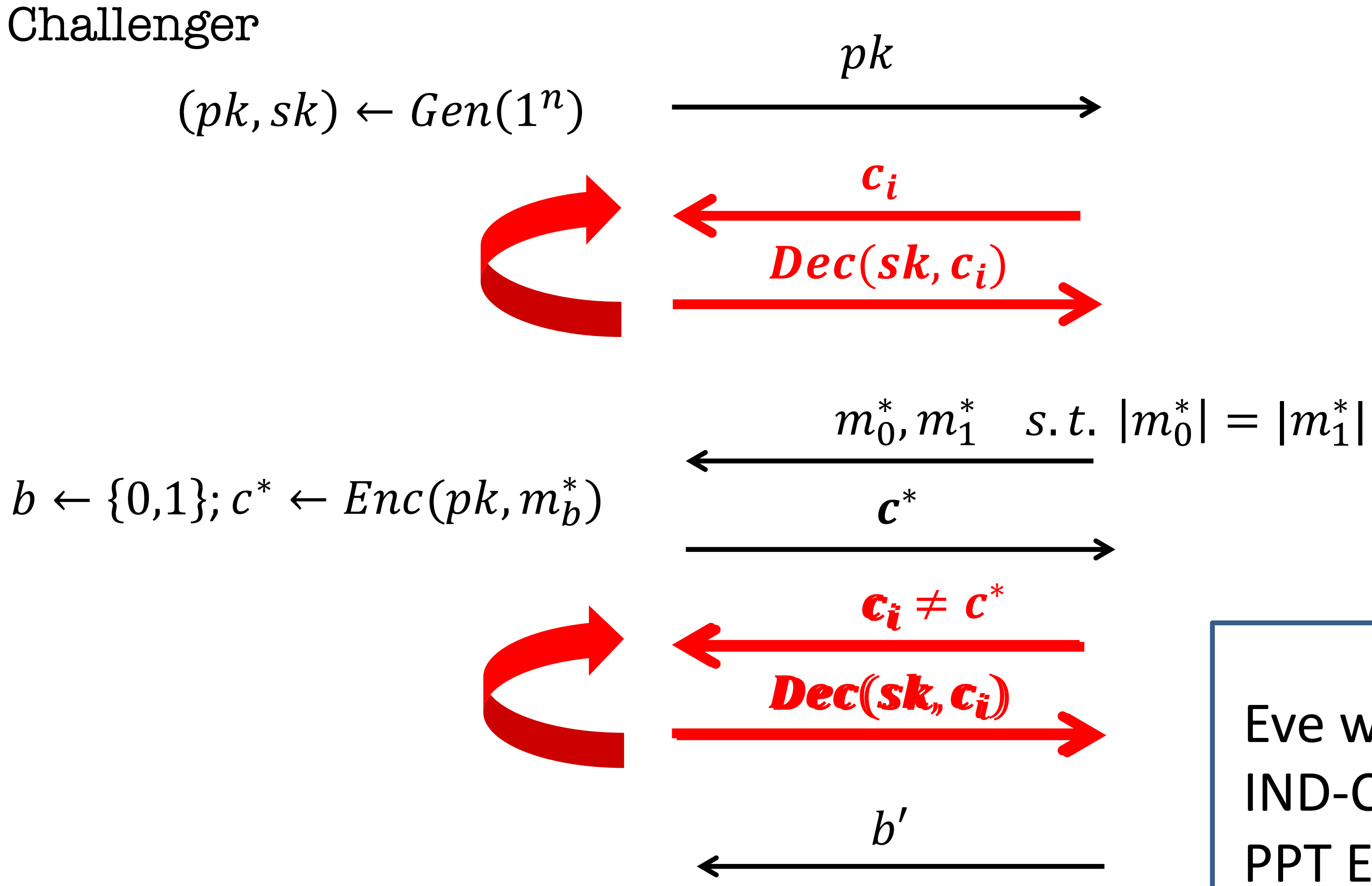
Active Attacks 2: Chosen-Ciphertext Attack



ATTACK: Adversary may have access to a decryption “oracle” and can use it to break security of a ”target” ciphertext c^* or even extract the secret key!

In fact, Bleichenbacher showed how to extract the entire secret key given only a “ciphertext verification” oracle.

IND-CCA Security



Challenger

Eve

$(pk, sk) \leftarrow Gen(1^n)$

pk

$\boldsymbol{c_i}$

$\boldsymbol{Dec(sk, c_i)}$

$m_0^*, m_1^* \quad s.t. \ |m_0^*| = |m_1^*|$

$b \leftarrow \{0,1\}; c^* \leftarrow Enc(pk, m_b^*)$

$\boldsymbol{c^*}$

$\boldsymbol{c_i \neq c^*}$

$\boldsymbol{Dec(sk, c_i)}$

b'

Eve wins if $b' = b$.
IND-CCA secure if no PPT Eve can win with prob. $> \frac{1}{2} + \mathrm{negl}(n)$.

Constructing CCA-Secure Encryption
(Intuition)

NIZK Proofs of Knowledge should help!

Idea: The encrypting party attaches an NIZK proof of knowledge of the underlying message to the ciphertext.

$C\colon\ (\mathrm{c} = \mathrm{CPAEnc}(m; r), \mathrm{proof}\ \pi\ that\ \text{“}I\ know\ m\ and\ r\text{”})$

This idea will turn out to be useful, but NIZK proofs themselves can be malleable!

Constructing CCA-Secure Encryption
(Intuition)

Digital Signatures should help!

OUR GOAL: Hard to modify an encryption of m into an encryption of a related message, say m+1.

Constructing CCA-Secure Encryption

Let's start with Digital Signatures.

$\boldsymbol{C}$: $(\mathbf{c} = \mathbf{CPAEnc}(\boldsymbol{pk}, \boldsymbol{m}; \boldsymbol{r}), Sign_{sgk}(c), vk)$

where the encryptor produces a signing / verification key pair by running $(sgk, vk) \leftarrow Sign.Gen(1^n)$

Is this CCA-secure/non-malleable?

If the adversary changes vk, all bets are off!

Lesson: NEED to "tie" the ciphertext c to vk in a "meaningful" way.

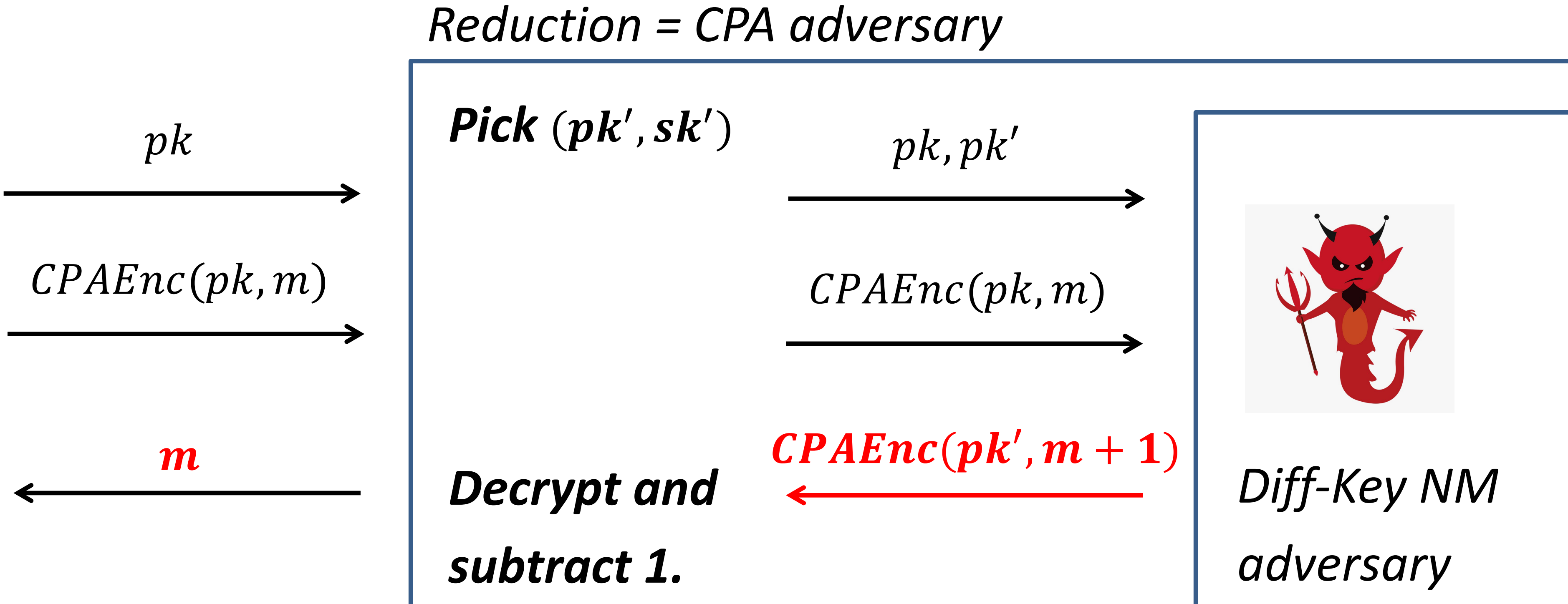
Observation: IND-CPA $\Longrightarrow$ “Different-Key Non-malleability”

Different-Key NM: Given $pk, pk', \mathrm{CPAEnc}(pk, m; r)$, **can an adversary produce** $\mathrm{CPAEnc}(pk', m+1; r)$?

NO! Suppose she could. Then, I can come up with a reduction that breaks the IND-CPA security of $\mathrm{CPAEnc}(pk, m; r)$.

Observation:
IND-CPA $\Longrightarrow$ "Different-Key Non-malleability"

Different-Key NM: Given $pk, pk', \mathrm{CPAEnc}(pk, m; r)$,
can an adversary produce $\mathrm{CPAEnc}(pk', m+1; r)$?



Reduction = CPA adversary

pk

$CPAEnc(pk, m)$

$\boldsymbol{m}$

Pick $(\boldsymbol{pk'}, \boldsymbol{sk'})$

pk, pk'

$CPAEnc(pk, m)$

$\boldsymbol{CPAEnc(pk', m+1)}$

Decrypt and subtract 1.

Diff-Key NM adversary

Putting it together

CCA Public Key: $2n$ public keys of the CPA scheme

(where $n = |vk|$)

$$\begin{bmatrix} pk_{1,0} & pk_{2,0} & \cdots & pk_{n,0} \\ pk_{1,1} & pk_{2,1} & & pk_{n,1} \end{bmatrix}$$

CCA Encryption:

First, pick a sign/ver key pair (sgk, vk)

$$CT = \begin{bmatrix} ct_{1,vk_1} & ct_{2,vk_2} & \cdots & ct_{n,vk_n} \end{bmatrix}$$

where $ct_{i,j} \leftarrow CPAEnc(pk_{i,j}, m)$

Output $(CT, vk, \sigma = Sign(sgk, CT))$.

Putting it together

Non-malleability rationale: Either

- Adversary keeps vk the same (in which case she has to break the signature scheme); or
- She changes the vk in which case she breaks the diff-NM game, and therefore CPA security.

CCA Encryption:

First, pick a sign/ver key pair (sgk, vk)

$$CT = \begin{bmatrix} ct_{1,vk_1} & ct_{2,vk_2} & \cdots & ct_{n,vk_n} \end{bmatrix}$$

where $ct_{i,j} \leftarrow CPAEnc(pk_{i,j}, m)$

Output $(CT, vk, \sigma = Sign(sgk, CT))$.

Call it a day?

We are not done!! Adversary could create ill-formed ciphertexts (e.g. the different cts encrypt different messages) and uses it for a Bleichenbacher-like attack.

CCA Encryption:

First, pick a sign/ver key pair (sgk, vk)

$$CT = \begin{bmatrix} ct_{1,vk_1} & ct_{2,vk_2} & \cdots & ct_{n,vk_n} \end{bmatrix}$$

where $ct_{i,j} \leftarrow CPAEnc(pk_{i,j}, m)$

Output $(CT, vk, \sigma = Sign(sgk, CT))$.

NIZK Proofs to the Rescue…

CCA Public Key: $2n$ public keys of the CPA scheme

$$\begin{bmatrix} pk_{1,0} & pk_{2,0} & \cdots & pk_{n,0} \\ pk_{1,1} & pk_{2,1} & & pk_{n,1} \end{bmatrix}, \textbf{\textit{CRS}}$$

NP statement: "there exist $m, r_{i,j}$ such that each $ct_{i,j} = CPAEnc(pk_{i,j}, m; r_{i,j})$"

key pair (sgk, vk)

$$\begin{bmatrix} \cdots & {}_{2,vk_2} & \cdots & ct_{n,vk_n} \end{bmatrix}$$

where $ct_{i,j} \leftarrow CPAEnc(pk_{i,j}, m; r_{i,j})$

π = NIZK proof that "CT is well-formed"

Output $(CT, \pi, vk, \sigma = Sign(sgk, (CT, \pi)))$.

Output $(CT, vk, \sigma = Sign(sgk, CT))$.

Are there other attacks?

Did we miss anything else?

Turns out NO. We can prove that this is CCA-secure.

For a proof sketch, see the next few slides and for a proof, read DDN.

We saw:
Non-Interactive Zero-Knowledge (NIZK) Proofs

We saw:
How to Construct CCA-secure encryption using NIZK proofs

Proof Sketch

Let’s play the CCA game with the adversary.

We will use her to break either the NIZK soundness/ZK, the signature scheme or the CPA-secure scheme.

Proof Sketch

Let’s play the CCA game with the adversary.

Hybrid 0: Play the CCA game as prescribed.

Hybrid 1: Observe that $\boldsymbol{vk_i} \neq \boldsymbol{vk}^*$.

(Otherwise break signature)

Observe that this means each query ciphertext-tuple involves a different public-key from the challenge ciphertext. Use the “different private-key” to decrypt.

(If the adv sees a difference, she broke NIZK soundness)

Hybrid 2: Now change the CRS/π into simulated CRS/π!

(OK by ZK)

If the Adv wins in this hybrid, she breaks **IND-CPA!**