Post-Quantum Proof of Knowledge from QLWE

Prabhanjan Ananth
UC Santa Barbara

Joint work with:
Kai-Min Chung (Academia Sinica, Taiwan),
Rolando L. La Placa (MIT $\rightarrow$ \$\$\$)

(to appear in CRYPTO'21)

Proof of Knowledge

Any classical prover who convinces a verifier to accept x must know a valid witness for x.

Preparing for Post-Quantum Era

THE WALL STREET JOURNAL.

Subscri

English Edition ▾ | Print Edition | Video | Podcasts | Latest Headlines

Home World U.S. Politics Economy Business Tech Markets Opinion Life & Arts Real Estate WSJ. Magazine Sports S

YOU HAVE BEEN SELECTED

WSJ wants to hear from you. Take part in this short survey to help shape The Journal. Click Here To Take The Survey

SHARE

CIO JOURNAL

Google Aims for Commercial-Grade Quantum Computer by 2029

Tech giant is one of many companies racing to build a business around the nascent technology

Google's new Quantum AI campus in Santa Barbara County, Calif., includes a quantum-data center, research labs and chip-fabrication facilities spanning several buildings.

PHOTO: ALPHABET

UPCOMING EVENTS

Jun 17 2021 — 12:00 PM - 1:45 PM EDT — WSJ Women In: Intelli Investing

Jun 24 2021 — 11:00 AM - 5:00 PM EDT — Global Food Forum

Jun 30 2021 — 1:00 PM - 1:45 PM EDT — WSJ Pro Cybersecurity Webinar: Aligning IT ar Cybersecurity

ADD TO CALENDAR

MOST POPULAR NEWS

1. National Parks Are Overcrowded and Closing Their Gates

Post-Quantum Proof of Knowledge (PQPoK) for NP [Unruh'12]

Any **quantum** *prover who convinces a verifier to accept x must know a valid witness for x.*

Difference between Classical Prover and Quantum Prover

- **Classical Prover**: intermediate states are binary strings.
- **Quantum Prover**: intermediate states are quantum states.

Difference between Classical Prover and Quantum Prover

- **Classical Prover**: intermediate states are binary strings.
- **Quantum Prover**: intermediate states are quantum states.

Rewinding a quantum prover is difficult!

Informal Definition of PQPoK for L

$\forall$ quantum prover $P^*, \exists$ black-box extractor $\mathcal{E}$,

Informal Definition of PQPoK for L

$\forall$ quantum prover P^*, $\exists$ black-box extractor $\mathcal{E}$,

- **Correctness of Extraction**:

$$\Pr\left[P^* \text{ convinces verifier to accept } x\right] = \varepsilon$$

$$\Rightarrow \Pr\left[(\rho_{\mathcal{E}}, w) \leftarrow \mathcal{E}^{P^*}(x) \bigwedge (x, w) \in \mathcal{R}(L)\right] = \varepsilon'$$

Informal Definition of PQPoK for L

$\forall$ quantum prover $P^*, \exists$ black-box extractor $\mathcal{E}$,

- **Correctness of Extraction**:

$$\Pr\left[P^* \text{ convinces verifier to accept } x\right] = \varepsilon$$

$$\Rightarrow \Pr\left[(\rho_{\mathcal{E}}, w) \leftarrow \mathcal{E}^{P^*}(x) \bigwedge (x, w) \in \mathcal{R}(L)\right] = \varepsilon'$$

P^* can be computationally unbounded.

Informal Definition of PQPoK for L

$\forall$ quantum prover P^*, $\exists$ black-box extractor $\mathcal{E}$,

- **Correctness of Extraction**:

$$\Pr\left[P^* \text{ convinces verifier to accept } x\right] = \varepsilon$$

$$\Rightarrow \Pr\left[(\rho_{\mathcal{E}}, w) \leftarrow \mathcal{E}^{P^*}(x) \bigwedge (x, w) \in \mathcal{R}(L)\right] = \varepsilon'$$

Ideally: $|\varepsilon' - \varepsilon| = \text{negl}$

Informal Definition of PQPoK for L

Another useful property:

- **Indistinguishability of Extraction**:

$$TD(\rho_V, \rho_{\mathcal{E}}) = \delta$$

(TD = Trace distance)

  - ρ_V: output state of P^* after interacting with V.
  - $\rho_{\mathcal{E}}$: output of $\mathcal{E}^{P^*}$.

Informal Definition of PQPoK for L

Another useful property:

- **Indistinguishability of Extraction**:

$$TD(\rho_V, \rho_{\mathcal{E}}) = \delta$$

(TD = Trace distance)

  - ρ_V: output state of P^* after interacting with V.
  - $\rho_{\mathcal{E}}$: output of $\mathcal{E}^{P^*}$.

Ideally: $\delta = \text{negl}$

Application: Secure Computation

Application: Secure Computation

Simulator uses the extractor to extract adversary's inputs

Application: Proof of Quantum Knowledge for QMA [Coladangelo-Vidick-Zhang'20]

Application: Proof of Quantum Knowledge for QMA [Coladangelo-Vidick-Zhang'20]

Prover$(x, |\Psi\rangle)$ **Verifier**(x)

$$\xrightarrow{X^a Z^b |\Psi\rangle}$$

$$\xleftrightarrow{\text{Post-Quantum PoK of } (a, b)}$$

Application: Proof of Quantum Knowledge for QMA [Coladangelo-Vidick-Zhang'20]

Prover$(x, |\Psi\rangle)$ **Verifier**(x)

$$\xrightarrow{X^a Z^b |\Psi\rangle}$$

$$\xleftrightarrow{\text{Post-Quantum PoK of } (a, b)}$$

Extractor can extract (a, b) and then recover $|\Psi\rangle$.

We use the fact here that (a, b) is classical.

First work on Post-Quantum PoK: [Unruh Eurocrypt'12]

Followups rely upon Unruh's technique.

Drawback with Unruh

Unruh's PQPoK does not satisfy
indistinguishability of extraction property.

Drawback with Unruh

Unruh's PQPoK does not satisfy indistinguishability of extraction property.

Prover's state after extraction

$$\not\approx$$

Prover's state after verifier's interaction.

Q: is it necessary for the extractor to disturb the prover's state in order to learn the witness?

Q: is it necessary for the extractor to disturb the prover's state in order to learn the witness?

At first glance, could seem inherent:

Q: is it necessary for the extractor to disturb the prover's state in order to learn the witness?

At first glance, could seem inherent:

Example: Prover could start with superposition of all the witnesses. If extractor learns w then prover's state should have collapsed to w.

Our Work

Theorem

Assuming LWE is hard against quantum polynomial-time algorithms,

There exists PQPoK for NP.

Techniques

Main Idea: Extraction via Oblivious Transfer

Main Idea: Extraction via Oblivious Transfer

Warmup: extraction of first bit of witness.

Extraction of first bit of witness

- Prover and Verifier run OT.

  Prover: role of sender

  Verifier: role of receiver.

Extraction of first bit of witness

- Prover and Verifier run OT.

  Prover: role of sender

  Verifier: role of receiver.

- Prover embeds w_1 (1st bit of witness) in one of the two locations at random.

Extraction of first bit of witness

- Prover and Verifier run OT.

  Prover: role of sender

  Verifier: role of receiver.

- Prover embeds w_1 (1st bit of witness) in one of the two locations at random.

- Verifier randomly guesses the location.

Extraction of first bit of witness

- Prover and Verifier run OT.
  Prover: role of sender
  Verifier: role of receiver.
- Prover embeds w_1 (1st bit of witness) in one of the two locations at random.
- Verifier randomly guesses the location.
- If the guessed location is correct, verifier gets w_1, o/w gets $\perp$.

First Attempt

Prover(x, w) **Verifier**(x)

.... OT phase ....

$b \xleftarrow{\$} \{0,1\}$

if $b = 0$, $(m_0, m_1) = (w_1, \perp)$

if $b = 1$, $(m_0, m_1) = (\perp, w_1)$ $b' \xleftarrow{\$} \{0,1\}$

Sender's input: (m_0, m_1) $\xleftrightarrow{\text{OT}}$ Receiver's input : b'

Prover(x, w) **Verifier**(x)

.... OT phase ....

$b \xleftarrow{\$} \{0, 1\}$

if $b = 0$, $(m_0, m_1) = (w_1, \perp)$

if $b = 1$, $(m_0, m_1) = (\perp, w_1)$ $b' \xleftarrow{\$} \{0, 1\}$

Sender's input: (m_0, m_1) $\xleftrightarrow{\text{OT}}$ Receiver's input : b'

.... Zero-Knowledge Phase ....

Prove $(x, w) \in L$ $\xleftrightarrow{\text{ZK}}$

and behaved honestly in OT

Requirement:
Post-Quantum ZK with soundness against unbounded quantum provers.

Extraction Process

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If recovered message is a bit, store 0 in register X. Otherwise store 1.

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If recovered message is a bit, store 0 in register X. Otherwise store 1.
- Measure X.

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If recovered message is a bit, store 0 in register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS,

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If recovered message is a bit, store 0 in register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **TBD**.

TBD step: *Can we perform Watrous rewinding?*

TBD step: *Can we perform Watrous rewinding?*

Watrous rewinding only works
IF the measurement outcome doesn't disturb the prover's state.

Input auxiliary state of P^* is $|\Psi\rangle = \frac{1}{\sqrt{2}}|\Psi_0\rangle|0\rangle_{\mathsf{A}} + \frac{1}{\sqrt{2}}|\Psi_1\rangle|1\rangle_{\mathsf{A}}$.

Input auxiliary state of P^* is $|\Psi\rangle = \frac{1}{\sqrt{2}}|\Psi_0\rangle|0\rangle_{\mathsf{A}} + \frac{1}{\sqrt{2}}|\Psi_1\rangle|1\rangle_{\mathsf{A}}$.

- If the value in register A is 0 then use $(\perp, \perp)$ in OT.
- If the value in register A is 1 then use (w_1, w_1) in OT.

Input auxiliary state of P^* is $|\Psi\rangle = \frac{1}{\sqrt{2}}|\Psi_0\rangle|0\rangle_{\mathsf{A}} + \frac{1}{\sqrt{2}}|\Psi_1\rangle|1\rangle_{\mathsf{A}}$.

- If the value in register A is 0 then use $(\perp, \perp)$ in OT.
- If the value in register A is 1 then use (w_1, w_1) in OT.

Input auxiliary state of P^* is $|\Psi\rangle = \frac{1}{\sqrt{2}}|\Psi_0\rangle|0\rangle_{\mathsf{A}} + \frac{1}{\sqrt{2}}|\Psi_1\rangle|1\rangle_{\mathsf{A}}$.

- If the value in register A is 0 then use $(\perp, \perp)$ in OT.
- If the value in register A is 1 then use (w_1, w_1) in OT.

- After rewinding, prover's state is $\approx |\Psi_1\rangle\langle\Psi_1|$.
- In the real world, prover's state is $\approx \frac{1}{2}|\Psi_0\rangle\langle\Psi_0| + \frac{1}{2}|\Psi_1\rangle\langle\Psi_1|$.

Input auxiliary state of P^* is $|\Psi\rangle = \frac{1}{\sqrt{2}}|\Psi_0\rangle|0\rangle_{\mathsf{A}} + \frac{1}{\sqrt{2}}|\Psi_1\rangle|1\rangle_{\mathsf{A}}$.

- If the value in register A is 0 then use $(\perp, \perp)$ in OT.
- If the value in register A is 1 then use (w_1, w_1) in OT.

- After rewinding, prover's state is $\approx |\Psi_1\rangle\langle\Psi_1|$.
- In the real world, prover's state is $\approx \frac{1}{2}|\Psi_0\rangle\langle\Psi_0| + \frac{1}{2}|\Psi_1\rangle\langle\Psi_1|$.

$$TD\left(|\Psi_1\rangle\langle\Psi_1|,\ \frac{1}{2}|\Psi_0\rangle\langle\Psi_0| + \frac{1}{2}|\Psi_1\rangle\langle\Psi_1|\right) \text{ is not small.}$$

Extraction Process

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If recovered message is a bit, store 0 in register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **TBD**.

FIX: modify the scheme to ensure that the measurement outcome does not affect the state.

Prover(x, w)		**Verifier**(x)

. . . . OT phase . . . .

$b \xleftarrow{\$} \{0,1\}$		
if $b = 0,\ (m_0, m_1) = (w_1, 0)$		
if $b = 1,\ (m_0, m_1) = (0, w_1)$		$b' \xleftarrow{\$} \{0,1\}$
Sender's input: (m_0, m_1)	$\xleftrightarrow{\text{OT}}$	Receiver's input : b'

. . . . Reveal Phase . . . .

	$\xrightarrow{\text{Reveal } b}$	

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If $b = b'$, store 0 in the register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **TBD**.

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If $b = b'$, store 0 in the register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **TBD**.

$$\Pr[b = b'] \approx \tfrac{1}{2} \text{ from OT security.}$$

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If $b = b'$, store 0 in the register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **TBD**.

$\Pr[b = b'] \approx \frac{1}{2}$ from OT security.

Since distribution of measurement outcomes is independent of aux state

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If $b = b'$, store 0 in the register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **TBD**.

$\Pr[b = b'] \approx \frac{1}{2}$ from OT security.

Since distribution of measurement outcomes is independent of aux state

$\Longrightarrow$

Measurement does not disturb the state.

Extractor rewinds ONLY IF the guessed location is different from b.

Extraction Process

Rewinding strategy:

- Run the prover P in superposition.
- If $b = b'$, store 0 in the register X. Otherwise store 1.
- Measure X.
- If the outcome is 0 then declare SUCCESS, otherwise **perform Watrous Rewinding**.

Why does checking $b = b'$ suffice?

Why does checking $b = b'$ suffice?

- **Case 1. Prover does not cheat**:
  Extractor extracts the first bit of the witness.

Why does checking $b = b'$ suffice?

- **Case 1. Prover does not cheat**:
  Extractor extracts the first bit of the witness.

- **Case 2. Prover cheats**:
  Extractor might have extracted garbage...

Why does checking $b = b'$ suffice?

- **Case 1. Prover does not cheat**:
  Extractor extracts the first bit of the witness.

- **Case 2. Prover cheats**:
  Extractor might have extracted garbage...
  ...but the prover will get caught in the ZK phase.

Protocol for extraction of 1st bit of witness

Prover(x, w) **Verifier**(x)

. . . . OT phase . . . .

$b \xleftarrow{\$} \{0, 1\}$

if $b = 0$, $(m_0, m_1) = (w_1, 0)$

if $b = 1$, $(m_0, m_1) = (0, w_1)$ $b' \xleftarrow{\$} \{0, 1\}$

Sender's input: (m_0, m_1) $\xleftrightarrow{\text{OT}}$ Receiver's input : b'

. . . . Reveal Phase . . . .

$\xrightarrow{\text{Reveal } b}$

. . . . Zero-Knowledge Phase . . . .

. . .

- Extractor extracts the first bit of witness

- Extractor extracts the first bit of witness
- ISSUE: Verifier can also recover the first bit of the witness with probability $\frac{1}{2}$

Error reduction

Error reduction

Error reduction

- Prover additively secret shares w_1 into $sh_1, \ldots, sh_\ell$.

Error reduction

- Prover additively secret shares w_1 into $sh_1, \ldots, sh_\ell$.
- Prover invokes ℓ instantiations of OT.

Error reduction

- Prover additively secret shares w_1 into $sh_1, \ldots, sh_\ell$.
- Prover invokes ℓ instantiations of OT.
- It embeds sh_i into the i^{th} instantiation of OT.

Prover$(x, w = w_1 \cdots w_\ell)$ **Verifier**(x)

$\ldots\ldots\ldots$ **Amplified OT for** w_1 $\ldots\ldots\ldots$

$\forall i, sh_i \xleftarrow{\$} \{0,1\} : \oplus_{i=1}^{\ell} sh_i = w_1$

Prover$(x, w = w_1 \cdots w_\ell)$ **Verifier**(x)

Amplified OT for w_1

$\forall i < \ell, sh_i \xleftarrow{\$} \{0,1\} : \oplus_{i=1}^{\ell} sh_i = w_1$

$b_1 \xleftarrow{\$} \{0,1\}$ $b'_1 \xleftarrow{\$} \{0,1\}$

Sender:$((1 - b_1) \cdot sh_1,\ b_1 \cdot sh_1)$ $\xleftrightarrow{\text{OT}}$ Receiver : b'_1

$\xrightarrow{\text{Reveal } b_1}$

Prover$(x, w = w_1 \cdots w_\ell)$		**Verifier**(x)

Amplified OT for w_1

$\forall i < \ell, sh_i \xleftarrow{\$} \{0,1\} : \oplus_{i=1}^{\ell} sh_i = w_1$		
$b_1 \xleftarrow{\$} \{0,1\}$		$b'_1 \xleftarrow{\$} \{0,1\}$
Sender:$((1-b_1)\cdot sh_1,\ b_1 \cdot sh_1)$	$\xleftrightarrow{\text{OT}}$	Receiver : b'_1
	$\xrightarrow{\text{Reveal } b_1}$	
	$\cdots$	
$b_\ell \xleftarrow{\$} \{0,1\}$		$b'_\ell \xleftarrow{\$} \{0,1\}$
Sender:$((1-b_\ell)\cdot sh_\ell,\ b_\ell \cdot sh_\ell)$	$\xleftrightarrow{\text{OT}}$	Receiver : b_ℓ
	$\xrightarrow{\text{Reveal } b_\ell}$	

So far: extraction of 1 bit of witness.

So far: extraction of 1 bit of witness.

Repeat this process for all the bits of the witness!

Prover(x, w) **Verifier**(x)

.... **OT phase** ....

$$\xleftrightarrow{\text{Amplified OT for } w_1}$$

$$\dots$$

$$\xleftrightarrow{\text{Amplified OT for } w_\ell}$$

...............Zero-Knowledge Phase...............

$$\dots$$

Instantiation of OT

Instantiation of OT

OT needs to have security against unbounded senders

Instantiation of OT

OT needs to have security against unbounded senders

Security against unbounded senders
$\Rightarrow$
security against unbounded P

Instantiation of OT

OT needs to have security against unbounded senders

Security against unbounded senders
$\Rightarrow$
security against unbounded P

Current known instantiations don't satisfy security against quantum poly-time receivers.

Construction of Post-Quantum OT

Goal: OT satisfying the following:

- Security against unbounded senders.
- Post-quantum security against receivers.

Construction of Post-Quantum OT

Goal: OT satisfying the following:

- Security against unbounded senders.
- Post-quantum security against receivers.

Start with OT satisfying:

- Security against unbounded receivers.
- Post-quantum security against senders.

Construction of Post-Quantum OT

Goal: OT satisfying the following:

- Security against unbounded senders.
- Post-quantum security against receivers.

Start with OT satisfying:

- Security against unbounded receivers.
- Post-quantum security against senders.

Construction from quantum hardness of LWE: [Brakerski-Döttling'18]

Start with OT satisfying:

- Security against unbounded receivers.
- Post-quantum security against senders.

Steps:

- OT reversal [WW'06,KKS'18,GJJM'20]

Start with OT satisfying:

- Security against unbounded receivers.
- Post-quantum security against senders.

Steps:

- OT reversal [WW'06,KKS'18,GJJM'20]

- Use a post-quantum statistical ZK: to prove correctness of OT.

Start with OT satisfying:

- Security against unbounded receivers.
- Post-quantum security against senders.

Steps:

- OT reversal [WW'06,KKS'18,GJJM'20]
- Use a post-quantum statistical ZK: to prove correctness of OT.

This protocol can be extended (painfully)
to the setting of bounded concurrent quantum ZK.

Summary

New construction of PQ PoK

Improves upon [Unruh'12]'s PQ PoK.

Summary

New construction of PQ PoK

Improves upon [Unruh'12]'s PQ PoK.

Thanks!