# Quantum security of subset cover problems

Samuel Bouaziz--Ermann
Joint work with Alex B. Grilo and Damien Vergnaud

LIP6, Sorbonne Université, CNRS

eprint:2022/1474 arXiv:2210.15396

June 20, 2023

# Motivation

Alice Bob

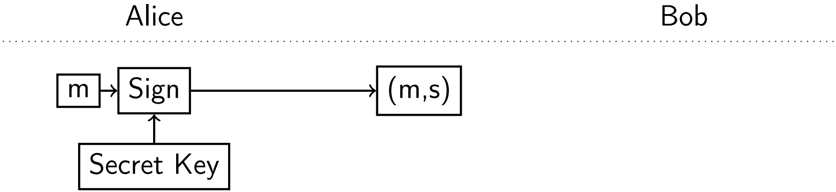
# Motivation

Alice Bob

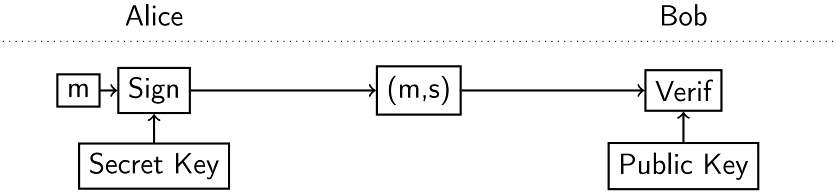
m

# Motivation

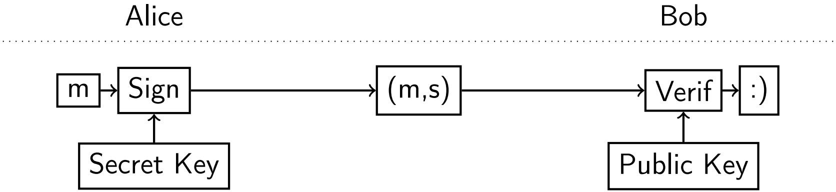
# Motivation



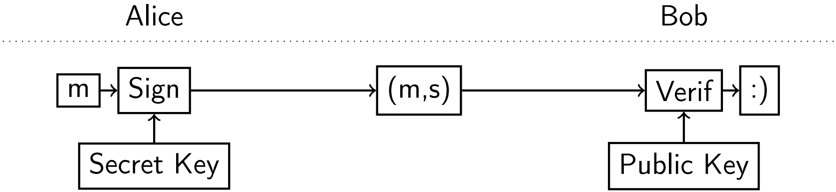
# Motivation



# Motivation



# Motivation




## Properties

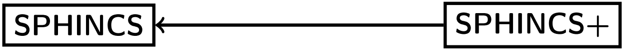
- Make sure that the message comes from Alice
- Verify the integrity of the message

# Motivation

SPHINCS+

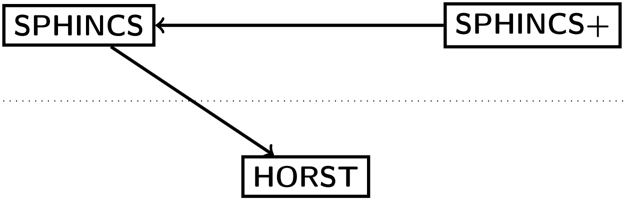
Post-quantum
signatures

# Motivation




Post-quantum signatures

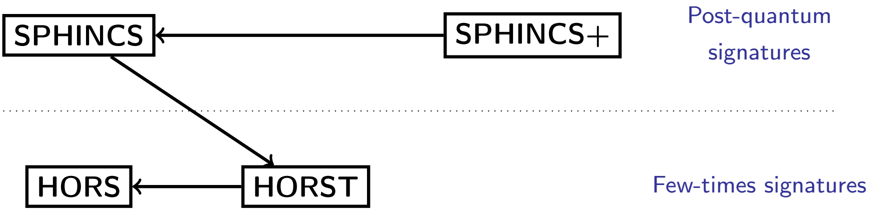
# Motivation




Post-quantum signatures

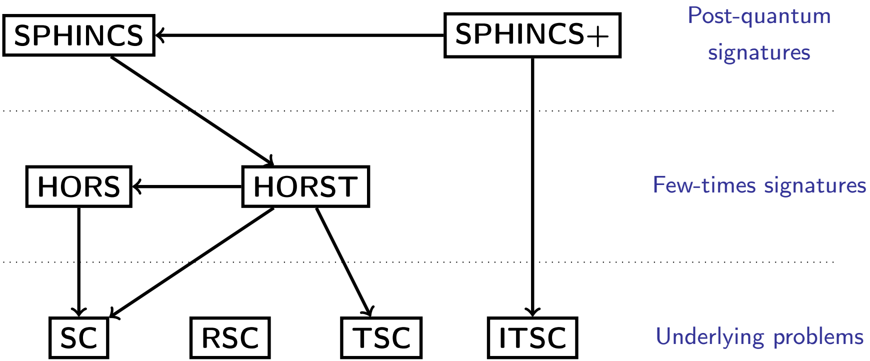
Few-times signatures

# Motivation

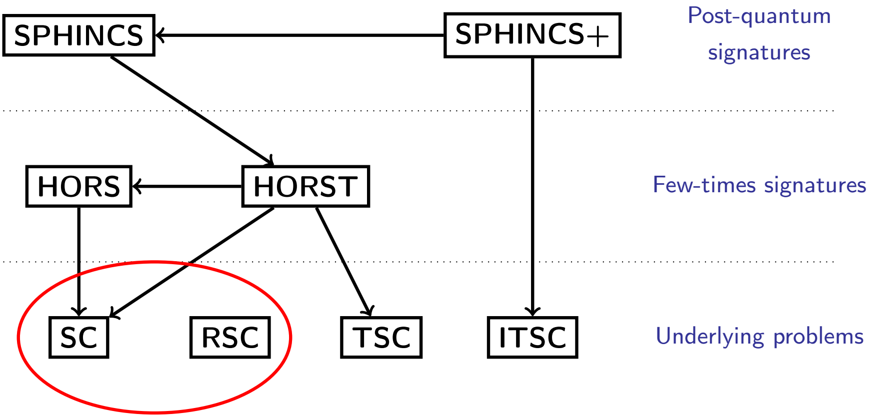


SPHINCS
SPHINCS+
Post-quantum signatures
HORS
HORST
Few-times signatures

# Motivation


SPHINCS
SPHINCS+
HORS
HORST
SC
RSC
TSC
ITSC
Post-quantum signatures
Few-times signatures
Underlying problems

# Motivation


SPHINCS
SPHINCS+
Post-quantum signatures
HORS
HORST
Few-times signatures
SC
RSC
TSC
ITSC
Underlying problems

### Relation between SPHINCS and the SC problem

## Relation between SPHINCS and the SC problem

- Finding a lower bound for the subset cover problem gives a lower bounds for the security of SPHINCS(+).

# Motivation

## Relation between SPHINCS and the SC problem

- Finding a lower bound for the subset cover problem gives a lower bounds for the security of SPHINCS(+).
- An algorithm that finds a subset cover **cannot** directly be used to attack SPHINCS(+).

# Motivation

## Relation between SPHINCS and the SC problem

- Finding a lower bound for the subset cover problem gives a lower bounds for the security of SPHINCS(+).
- An algorithm that finds a subset cover **cannot** directly be used to attack SPHINCS(+).
- The subset cover variations (RSC, TSC, and ITSC) are also linked to the security of SPHINCS and SPHINCS+.

# Context

## Definition

- A **hash function** is a function $h : \{0, 1\}^* \to \{0, 1\}^n$.

# Context

## Definition

- A **hash function** is a function $h : \{0,1\}^* \rightarrow \{0,1\}^n$.
- In the **Random Oracle Model** (ROM), a hash function is modelized as a *random* function $\mathcal{H} : \mathcal{X} \rightarrow \mathcal{Y}$.

## Definition

- A **hash function** is a function $h : \{0,1\}^* \to \{0,1\}^n$.
- In the **Random Oracle Model** (ROM), a hash function is modelized as a *random* function $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$.
- In the **Quantum Random Oracle Model** (QROM), the random function $\mathcal{H}$ can be queried in superposition:

$$O\left(\sum_{x,y} \alpha_{x,y} |x\rangle |y\rangle\right) = \sum_{x,y} \alpha_{x,y} |x\rangle |y + O(x)\rangle$$

# Context

## Definition

- A **hash function** is a function $h : \{0,1\}^* \to \{0,1\}^n$.
- In the **Random Oracle Model** (ROM), a hash function is modelized as a *random* function $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$.
- In the **Quantum Random Oracle Model** (QROM), the random function $\mathcal{H}$ can be queried in superposition:

$$O\left(\sum_{x,y} \alpha_{x,y} |x\rangle |y\rangle\right) = \sum_{x,y} \alpha_{x,y} |x\rangle |y + O(x)\rangle$$

## Our model

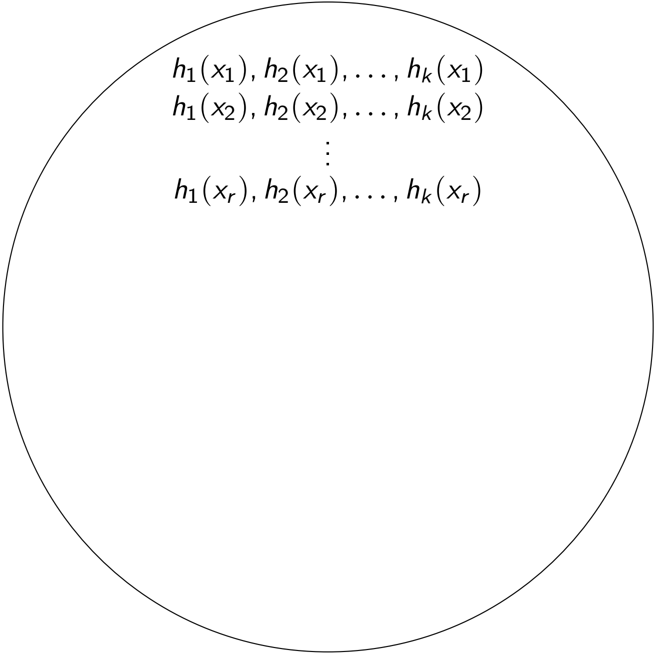
In our model, we have quantum access to $h_1, \cdots, h_k$ such that $h_i : \mathcal{X} \to \mathcal{Y}$, and $|\mathcal{Y}| = N$.

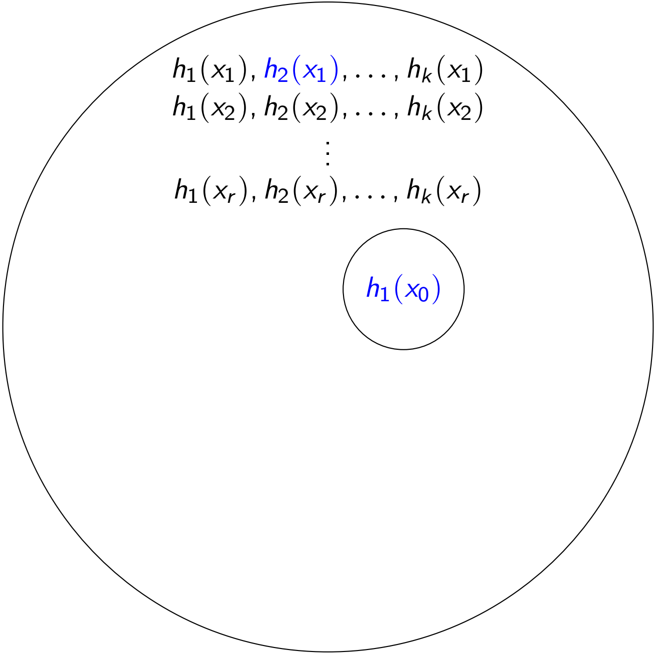
## Definition

A $(r, k)$-subset cover ($(r, k)$-SC) for $h_1, \ldots, h_k$ consist of $r + 1$ elements $x_0, x_1, \ldots, x_r$ in domain $\mathcal{X}$ such that:

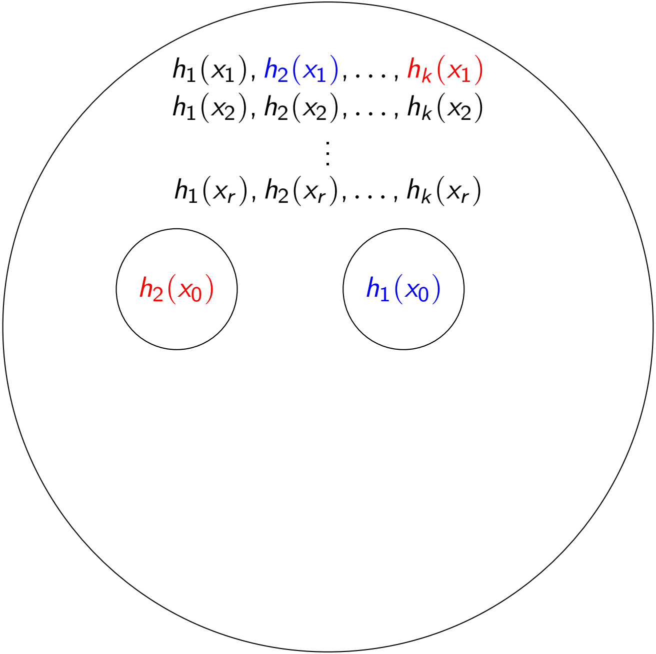
$$x_0 \notin \{x_1, \ldots, x_r\}, \text{ and}$$

$$\{h_i(x_0) | 1 \leq i \leq k\} \subseteq \bigcup_{j=0}^{r} \{h_i(x_j) | 1 \leq i \leq k\}.$$

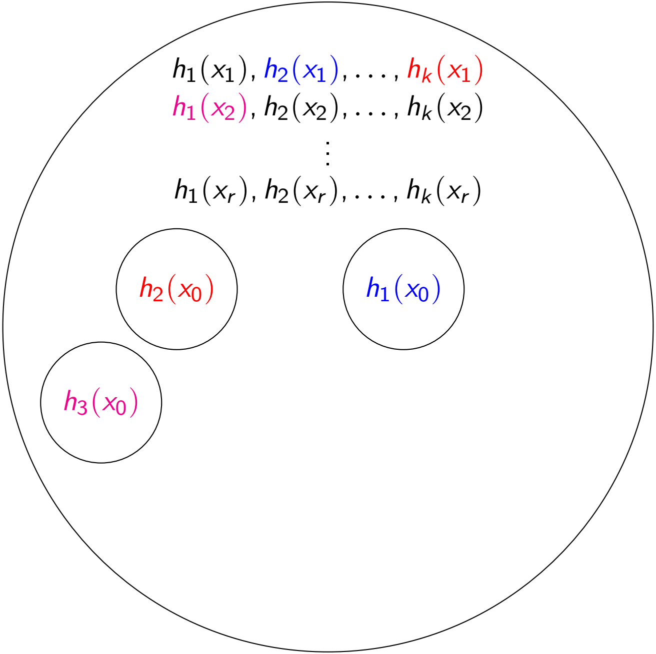

$h_1(x_1), h_2(x_1), \ldots, h_k(x_1)$
$h_1(x_2), h_2(x_2), \ldots, h_k(x_2)$
$\vdots$
$h_1(x_r), h_2(x_r), \ldots, h_k(x_r)$


$h_1(x_1), h_2(x_1), \ldots, h_k(x_1)$
$h_1(x_2), h_2(x_2), \ldots, h_k(x_2)$
$\vdots$
$h_1(x_r), h_2(x_r), \ldots, h_k(x_r)$
$h_1(x_0)$

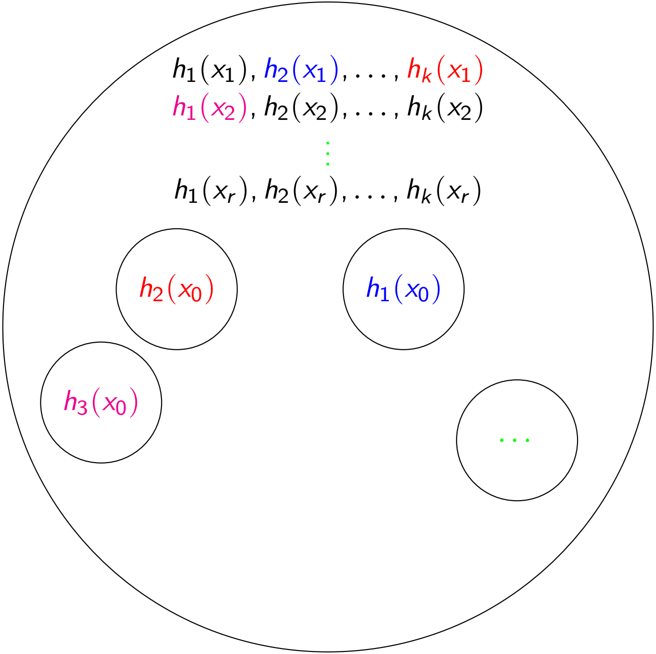
# Subset Cover



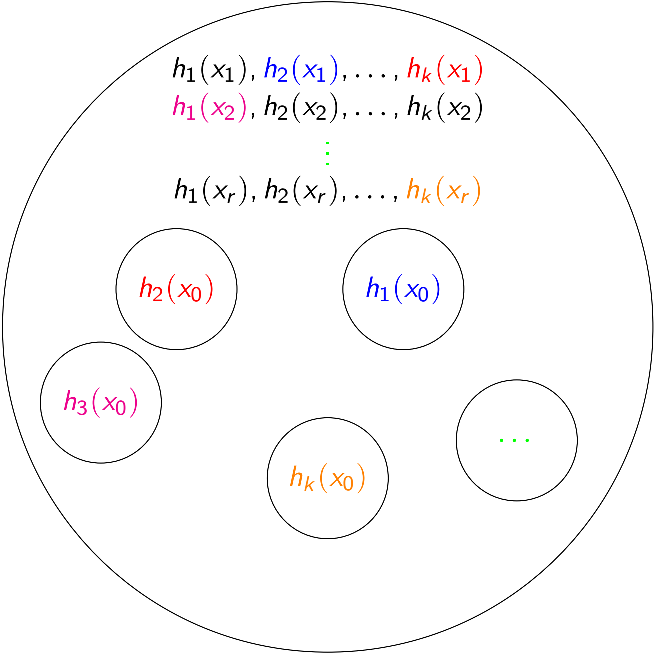
# Subset Cover



# Subset Cover



## Subset Cover



## Definition

A $k$-restricted subset cover ($k$–RSC) for $h_1, \ldots, h_k$ consist of $k+1$ elements $x_0, x_1, \ldots, x_k$ in domain $\mathcal{X}$ such that:

$$x_0 \notin \{x_1, \ldots, x_k\}, \text{ and}$$

$$\begin{aligned} h_1(x_0) &= h_1(x_1), \\ h_2(x_0) &= h_2(x_2), \\ &\vdots \\ h_k(x_0) &= h_k(x_k), \end{aligned}$$

# Results on subset cover

**Theorem ([YTA22])**

*There exists an algorithm that find a k–RSC by making* $O\left(k \cdot N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ *queries to* $h_1, \ldots, h_k$.

# Results on subset cover

**Theorem ([YTA22])**

*There exists an algorithm that find a $k$–RSC by making $O\left(k \cdot N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ queries to $h_1, \ldots, h_k$.*

No other work!

# Our results on $k$–RSC

## Lower bound on $k$–RSC

We prove that $\Omega\left(k^{-\frac{2^k}{2^{k+1}-1}} \cdot N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ quantum queries to the idealized hash functions are needed to find a $k$–RSC with constant probability.

## Lower bound on $k$–RSC

We prove that $\Omega\left(k^{-\frac{2^k}{2^{k+1}-1}} \cdot N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ quantum queries to the idealized hash functions are needed to find a $k$–RSC with constant probability.

## Lower bound on 2–RSC

Given two random functions $h_1, h_2 : \mathcal{X} \to \mathcal{Y}$ where $|N| = \mathcal{Y}$, a quantum algorithm needs to make $\Omega(N^{3/7})$ queries to $h_1$ and $h_2$ to find a 2–RSC with a constant probability.

# Quantum information

- We consider elements of a Hilbert space $\mathcal{H}$ of dimension $2^n$.

- We consider elements of a Hilbert space $\mathcal{H}$ of dimension $2^n$.
- We write $|0\rangle, |1\rangle, \cdots, |2^n - 1\rangle$ the computational basis.

- We consider elements of a Hilbert space $\mathcal{H}$ of dimension $2^n$.
- We write $|0\rangle, |1\rangle, \cdots, |2^n - 1\rangle$ the computational basis.
- A quantum state $|\phi\rangle$ is an element of $\mathcal{H}$ of norm 1, and can be written:

$$|\phi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle .$$

- We consider elements of a Hilbert space $\mathcal{H}$ of dimension $2^n$.
- We write $|0\rangle, |1\rangle, \cdots, |2^n - 1\rangle$ the computational basis.
- A quantum state $|\phi\rangle$ is an element of $\mathcal{H}$ of norm 1, and can be written:

$$|\phi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle .$$

- We write $\langle\phi|$ the conjugate transpose of $|\phi\rangle$.

- We consider elements of a Hilbert space $\mathcal{H}$ of dimension $2^n$.
- We write $|0\rangle, |1\rangle, \cdots, |2^n - 1\rangle$ the computational basis.
- A quantum state $|\phi\rangle$ is an element of $\mathcal{H}$ of norm 1, and can be written:

$$|\phi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle .$$

- We write $\langle\phi|$ the conjugate transpose of $|\phi\rangle$.
- We can apply unitaries $U$ to quantum state, and this corresponds to doing computation.

# Quantum information

- We consider elements of a Hilbert space $\mathcal{H}$ of dimension $2^n$.
- We write $|0\rangle, |1\rangle, \cdots, |2^n - 1\rangle$ the computational basis.
- A quantum state $|\phi\rangle$ is an element of $\mathcal{H}$ of norm 1, and can be written:

$$|\phi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle .$$

- We write $\langle\phi|$ the conjugate transpose of $|\phi\rangle$.
- We can apply unitaries $U$ to quantum state, and this corresponds to doing computation.
- We can measure quantum state, doing so give $i$ with probability $|\alpha_i|^2$.

## QFT

The **Quantum Fourier Transform** is a unitary that, given an input state $|k\rangle$, outputs:

$$\frac{1}{2^{n/2}} \sum_{\ell=0}^{2^n-1} \omega_n^{k\ell} |k\rangle ,$$

where $\omega_n = e^{2\pi i/2^n}$.

## QFT

The **Quantum Fourier Transform** is a unitary that, given an input state $|k\rangle$, outputs:

$$\frac{1}{2^{n/2}} \sum_{\ell=0}^{2^n-1} \omega_n^{k\ell} |k\rangle ,$$

where $\omega_n = e^{2\pi i/2^n}$.

- This unitary can be efficiently implemented, and we write it *QFT*.

# Quantum Fourier Transform

## QFT

The **Quantum Fourier Transform** is a unitary that, given an input state $|k\rangle$, outputs:

$$\frac{1}{2^{n/2}} \sum_{\ell=0}^{2^n-1} \omega_n^{k\ell} |k\rangle ,$$

where $\omega_n = e^{2\pi i/2^n}$.

- This unitary can be efficiently implemented, and we write it *QFT*.
- Applying the *QFT* to the computational basis $|0\rangle, \ldots, |N\rangle$ yields the *Fourier basis* $|\widehat{0}\rangle, \ldots, \left|\widehat{N}\right\rangle$.

# Quantum Fourier Transform

## QFT

The **Quantum Fourier Transform** is a unitary that, given an input state $|k\rangle$, outputs:

$$\frac{1}{2^{n/2}} \sum_{\ell=0}^{2^n-1} \omega_n^{k\ell} |k\rangle ,$$

where $\omega_n = e^{2\pi i/2^n}$.

- This unitary can be efficiently implemented, and we write it *QFT*.
- Applying the *QFT* to the computational basis $|0\rangle, \ldots, |N\rangle$ yields the *Fourier basis* $|\widehat{0}\rangle, \ldots, |\widehat{N}\rangle$.
- In particular, we have that:

$$|\widehat{0}\rangle = \frac{1}{2^{n/2}} \sum_{\ell=0}^{2^n-1} |\ell\rangle .$$

# Compressed Random Oracle

- When making a query in the QROM:

$$\mathrm{StO}\left(\sum_{x,y} \alpha_{x,y} |x\rangle |y\rangle\right) = \sum_{x,y} \alpha_{x,y} |x\rangle |y + O(x)\rangle$$

- When making a query in the QROM:

$$\text{StO}\left(\sum_{x,y}\alpha_{x,y}\left|x\right\rangle\left|y\right\rangle\right)=\sum_{x,y}\alpha_{x,y}\left|x\right\rangle\left|y+O(x)\right\rangle$$

- We add a new register, for the database: $|D\rangle$. We write $\mathcal{D}$ the set of all possible database.

# Compressed Random Oracle

- When making a query in the QROM:

$$\mathrm{StO}\left(\sum_{x,y} \alpha_{x,y} |x\rangle |y\rangle\right) = \sum_{x,y} \alpha_{x,y} |x\rangle |y + O(x)\rangle$$

- We add a new register, for the database: $|D\rangle$. We write $\mathcal{D}$ the set of all possible database.
- $|D\rangle$ is initially equal the uniform supersition:

$$\sum_{D \in \mathcal{D}} \frac{1}{|\mathcal{D}|} |D\rangle$$

- When making a query in the QROM:

$$\mathsf{StO}\left(\sum_{x,y}\alpha_{x,y}\left|x\right\rangle\left|y\right\rangle\right)=\sum_{x,y}\alpha_{x,y}\left|x\right\rangle\left|y+O(x)\right\rangle$$

- We add a new register, for the database: $|D\rangle$. We write $\mathcal{D}$ the set of all possible database.
- $|D\rangle$ is initially equal the uniform supersition:

$$\sum_{D\in\mathcal{D}}\frac{1}{|\mathcal{D}|}\left|D\right\rangle$$

- When we make a query:

$$\mathsf{O}\left(\sum_{x,y,D}\alpha_{x,y,D}\left|x\right\rangle\left|y\right\rangle\left|D\right\rangle\right)=\sum_{x,y,D}\alpha_{x,y,D}\left|x\right\rangle\left|y\right\rangle\left|D\oplus(x,y)\right\rangle$$

### Compression part

We apply the compression operator in the Fourier basis

$$\mathrm{Comp} = \bigotimes_{x} \left( |\perp\rangle \langle \hat{0}| + \sum_{y:y \neq \hat{0}} |y\rangle \langle y| \right),$$

where $\perp$ is a new symbol and $\left|\hat{0}\right\rangle = \sum_{i=0}^{2^n-1} \frac{1}{2^n} |i\rangle$ is the uniform superposition.

# Compressed Random Oracle

## Compression part

We apply the compression operator in the Fourier basis

$$\mathrm{Comp} = \bigotimes_x \left( |\perp\rangle \langle \hat{0}| + \sum_{y : y \neq \hat{0}} |y\rangle \langle y| \right),$$

where $\perp$ is a new symbol and $\left|\hat{0}\right\rangle = \sum_{i=0}^{2^n-1} \frac{1}{2^n} |i\rangle$ is the uniform superposition.

## The full Oracle

$$\mathrm{cO} = \mathrm{Comp} \circ \mathrm{O} \circ \mathrm{Comp}^\dagger$$

# Compressed Random Oracle

## Description

After $q$ queries the state of the database can be described with $q$ vectors.

## Lemma (Zhandry)

*Let $A$ be an algorithm that makes $k$ queries to a random oracle $H : \mathcal{X} \to \mathcal{Y}$. Then, the compressed random oracle technique simulates $H$ with an error at most $\sqrt{\frac{k}{|\mathcal{Y}|}}$.*

This essentially corresponds to on-the-fly simulation of $H$.

## Database properties

- A database property $P$ is a subset of $\mathcal{D}$.
- For example, $P_{pre-image} = \{D | \exists x \in D, H(x) = 0\}$.
- Any database property $P$ can be seen as a projector on $\mathcal{D}$, as follows:

$$\sum_{d \in P} |d\rangle \langle d|$$

# Using CROM to compute lower bounds

## Database properties

- A database property $P$ is a subset of $\mathcal{D}$.
- For example, $P_{pre-image} = \{D | \exists x \in D, H(x) = 0\}$.
- Any database property $P$ can be seen as a projector on $\mathcal{D}$, as follows:

$$\sum_{d \in P} |d\rangle \langle d|$$

## Computing lower bounds

If we write $|\phi_i\rangle = cOU_q cOU_{q-1} \dots cOU_1 |0\rangle^{\otimes n}$ the state of an algorithm $A$ after $i$ queries to the oracle, we want to bound:

$$|P |\phi_i\rangle | \leq f(i)$$

Thus $f^{-1}(i)$ queries to $H$ are necessary to find a database that satisfies property $P$.

# Sketch of the proof

## Lower bound on 2–RSC

Given two random functions $h_1, h_2 : \mathcal{X} \to \mathcal{Y}$ where $|N| = \mathcal{Y}$, a quantum algorithm needs to make $\Omega(N^{3/7})$ queries to $h_1$ and $h_2$ to find a 2–RSC with a constant probability.

# Sketch of the proof

## Lower bound on 2–RSC

Given two random functions $h_1, h_2 : \mathcal{X} \to \mathcal{Y}$ where $|N| = \mathcal{Y}$, a quantum algorithm needs to make $\Omega(N^{3/7})$ queries to $h_1$ and $h_2$ to find a 2–RSC with a constant probability.

## Database properties

- $P_2$ as the set of databases that contain a 2-RSC.

# Sketch of the proof

## Lower bound on 2–RSC

Given two random functions $h_1, h_2 : \mathcal{X} \to \mathcal{Y}$ where $|N| = \mathcal{Y}$, a quantum algorithm needs to make $\Omega(N^{3/7})$ queries to $h_1$ and $h_2$ to find a 2–RSC with a constant probability.

## Database properties

- $P_2$ as the set of databases that contain a 2-RSC.
- $P'_\ell$ as the set of databases that contain *at least* $\ell$ *distinct* collisions on $h_1$.

# Sketch of the proof

## Lower bound on 2–RSC

Given two random functions $h_1, h_2 : \mathcal{X} \to \mathcal{Y}$ where $|N| = \mathcal{Y}$, a quantum algorithm needs to make $\Omega(N^{3/7})$ queries to $h_1$ and $h_2$ to find a 2–RSC with a constant probability.

## Database properties

- $P_2$ as the set of databases that contain a 2-RSC.
- $P'_\ell$ as the set of databases that contain *at least* $\ell$ *distinct* collisions on $h_1$.

Writing $|\psi_i\rangle$ the state after the $i^{th}$ query to $H = (h_1, h_2)$, our goal is to bound $|P_2 |\psi_i\rangle|$.

# Sketch of the proof

## Claim

We have that:

$$\left| P_2 \left| \psi_i \right\rangle \right| \leq \left| P_2 \left| \psi_{i-1} \right\rangle \right| + \left| P_2 c O (I - P_2) \left| \psi_{i-1} \right\rangle \right| .$$

# Sketch of the proof

## Claim

We have that:

$$|P_2 |\psi_i\rangle| \leq |P_2 |\psi_{i-1}\rangle| + |P_2 cO(I - P_2) |\psi_{i-1}\rangle)| .$$

$$|P_2 cO(I - P_2) |\psi_{i-1}\rangle|$$

$$= \left| P_2 cO \sum_{\substack{x,y \\ D\text{: no 2--RSC}}} \alpha_{x,y,D} |x, y, D\rangle \right|$$

# Sketch of the proof

## Claim

We have that:

$$|P_2 \left|\psi_i\right\rangle| \leq |P_2 \left|\psi_{i-1}\right\rangle| + |P_2 cO(I - P_2) \left|\psi_{i-1}\right\rangle)| .$$

$$\begin{aligned}
&|P_2 cO(I - P_2) \left|\psi_{i-1}\right\rangle| \\
&= \left| P_2 cO \sum_{\substack{x,y \\ D:\text{ no 2–RSC}}} \alpha_{x,y,D} \left|x, y, D\right\rangle \right| \\
&= \left| P_2 \sum_{\substack{x,y \\ D:\text{ no 2–RSC}}} \frac{1}{\sqrt{N^2}} \sum_{y'} \omega_n^{yy'} \alpha_{x,y,D} \left|x, y, D \oplus (x, y')\right\rangle \right|
\end{aligned}$$

# Sketch of the proof

**Claim**

We have that:

$$|P_2 |\psi_i\rangle| \leq |P_2 |\psi_{i-1}\rangle| + |P_2 cO(I - P_2) |\psi_{i-1}\rangle|\,.$$

$$\begin{aligned}
&|P_2 cO(I - P_2) |\psi_{i-1}\rangle| \\
&= \left| P_2 cO \sum_{\substack{x,y \\ D:\text{ no 2–RSC}}} \alpha_{x,y,D} |x, y, D\rangle \right| \\
&= \left| P_2 \sum_{\substack{x,y \\ D:\text{ no 2–RSC}}} \frac{1}{\sqrt{N^2}} \sum_{y'} \omega_n^{yy'} \alpha_{x,y,D} |x, y, D \oplus (x, y')\rangle \right| \\
&= \left| \sum_{y'} \sum_{\substack{x,y \\ D:\text{ no 2–RSC}}} \frac{\omega_n^{yy'} \alpha_{x,y,D}}{\sqrt{N^2}} \cdot P_2 |x, y, D \oplus (x, y')\rangle \right|.
\end{aligned}$$

# Sketch of the proof

### 2-RSC

Recall that a 2–RSC consist of $x_0, x_1, x_2$ such that:

- $h_1(x_0) = h_1(x_1)$
- $h_2(x_0) = h_2(x_2)$

# Sketch of the proof

## 2-RSC

Recall that a 2–RSC consist of $x_0, x_1, x_2$ such that:

- $h_1(x_0) = h_1(x_1)$
- $h_2(x_0) = h_2(x_2)$

We have three possible ways to get from $D$ that does not have a 2–RSC to $D \oplus (x, y')$ that has a 2–RSC:

- $x = x_2$ or $x = x_1$
- $x = x_0$

# Sketch of the proof

## 2-RSC

Recall that a 2–RSC consist of $x_0, x_1, x_2$ such that:

- $h_1(x_0) = h_1(x_1) \leftarrow$ There must be a collision in the database
- $h_2(x_0) = h_2(x_2) \leftarrow$ We need a new collision there

We have three possible ways to get from $D$ that does not have a 2–RSC to $D \oplus (x, y')$ that has a 2–RSC:

- $x = x_2$ or $x = x_1$
- $x = x_0$

# Sketch of the proof

## 2-RSC

Recall that a 2–RSC consist of $x_0, x_1, x_2$ such that:

- $h_1(x_0) = h_1(x_1) \leftarrow$ There must be a collision in the database
- $h_2(x_0) = h_2(x_2) \leftarrow$ We need a new collision there

We have three possible ways to get from $D$ that does not have a 2–RSC to $D \oplus (x, y')$ that has a 2–RSC:

- $x = x_2$ or $x = x_1$
- $x = x_0$

If there are $\ell$ collisions on $h_1$, there are $\ell$ values of $y'$ such that $D \oplus (x, y')$ has a 2–RSC.

# Sketch of the proof

## 2-RSC

Recall that a 2–RSC consist of $x_0, x_1, x_2$ such that:

- $h_1(x_0) = h_1(x_1) \leftarrow$ We need a new collision there
- $h_2(x_0) = h_2(x_2) \leftarrow$ We need a new collision there

We have three possible ways to get from $D$ that does not have a 2–RSC to $D \oplus (x, y')$ that has a 2–RSC:

- $x = x_2$ or $x = x_1$
- $x = x_0$

# Sketch of the proof

## 2-RSC

Recall that a 2–RSC consist of $x_0, x_1, x_2$ such that:

- $h_1(x_0) = h_1(x_1) \leftarrow$ We need a new collision there
- $h_2(x_0) = h_2(x_2) \leftarrow$ We need a new collision there

We have three possible ways to get from $D$ that does not have a 2–RSC to $D \oplus (x, y')$ that has a 2–RSC:

- $x = x_2$ or $x = x_1$
- $x = x_0$

There are $(i-1)^2$ values of $y'$ such that $D \oplus (x, y')$ has a 2–RSC.

# Sketch of the proof

Thus,

$$|P_2 cO(I - P_2)\,|\psi_{i-1}\rangle| \leq \sqrt{2 \cdot \sum_{\ell \geq 0} \frac{\ell}{N} \left|P'_\ell\,|\psi_{i-1}\rangle\right|^2 + \frac{i-1}{N}}$$

Thus,

$$\left|P_2 cO(I - P_2)\left|\psi_{i-1}\right\rangle\right| \leq \sqrt{2 \cdot \sum_{\ell \geq 0} \frac{\ell}{N} \left|P'_\ell \left|\psi_{i-1}\right\rangle\right|^2} + \frac{i-1}{N}$$

Giving:

$$\left|P_2 \left|\psi_i\right\rangle\right| \leq \left|P_2 \left|\psi_{i-1}\right\rangle\right| + \sqrt{2 \sum_{\ell \geq 0} \frac{\ell}{N} \left|P'_\ell \left|\psi_{i-1}\right\rangle\right|^2} + \frac{i-1}{N}$$

Thus,

$$|P_2 cO(I - P_2)\,|\psi_{i-1}\rangle| \leq \sqrt{2 \cdot \sum_{\ell \geq 0} \frac{\ell}{N} \left|P'_\ell\,|\psi_{i-1}\rangle\right|^2} + \frac{i-1}{N}$$

Giving:

$$\begin{aligned} |P_2\,|\psi_i\rangle| &\leq |P_2\,|\psi_{i-1}\rangle| + \sqrt{2 \sum_{\ell \geq 0} \frac{\ell}{N} \left|P'_\ell\,|\psi_{i-1}\rangle\right|^2} + \frac{i-1}{N} \\ &\leq \dots \\ &\leq \sum_{s=0}^{i-1} \left( \sqrt{2 \sum_{\ell \geq 0} \frac{\ell}{N} \left|P'_\ell\,|\psi_s\rangle\right|^2} + \frac{\ell}{N} \right). \end{aligned}$$

# Sketch of the proof

**Lemma**

*For every $i \in \mathbb{N}$, we have that:*

$$|P_2 \,|\psi_i\rangle| \leq 4\sqrt{e}\frac{i^{7/4}}{N^{3/4}} + 4\frac{i^2}{N} + O(N^{-1/48}).$$

**Lemma**

*For every $i \in \mathbb{N}$, we have that:*

$$|P_2 |\psi_i\rangle| \leq 4\sqrt{e}\frac{i^{7/4}}{N^{3/4}} + 4\frac{i^2}{N} + O(N^{-1/48}).$$

So when $i = o(N^{3/7})$, we have $g_i = o(1)$. Hence if we want $g_i$ no to be constant, i.e. not $o(1)$, we must have $i = \Omega\left(N^{3/7}\right)$.

# Sketch of the proof

**Lemma**

*For every $i \in \mathbb{N}$, we have that:*

$$|P_2 |\psi_i\rangle| \leq 4\sqrt{e}\frac{i^{7/4}}{N^{3/4}} + 4\frac{i^2}{N} + O(N^{-1/48}).$$

So when $i = o(N^{3/7})$, we have $g_i = o(1)$. Hence if we want $g_i$ no to be constant, i.e. not $o(1)$, we must have $i = \Omega\left(N^{3/7}\right)$.

**Lower bound on 2–RSC**

Given two random functions $h_1, h_2 : \mathcal{X} \to \mathcal{Y}$ where $|N| = \mathcal{Y}$, a quantum algorithm needs to make $\Omega(N^{3/7})$ queries to $h_1$ and $h_2$ to find a 2–RSC with a constant probability.

# Results on $(r, k)$–SC

## Lower bound on $(1, k)$–SC

We prove that $\Omega\left((k!)^{-1/5} \cdot N^{k/5}\right)$ quantum queries to the idealized hash functions are needed to find a $(1, k)$–SC with constant probability.

# Results on $(r, k)$–SC

## Lower bound on $(1, k)$–SC

We prove that $\Omega\left((k!)^{-1/5} \cdot N^{k/5}\right)$ quantum queries to the idealized hash functions are needed to find a $(1, k)$–SC with constant probability.

## Upper bound on $(r, k)$–SC

We design an algorithm that finds a $(r, k)$–SC with constant probability by making $O\left(N^{k/(2+2r)}\right)$ queries to the hash functions.

### Definition (Search problem)

We are given a function $F : \mathcal{X} \to \{0,1\}$. The search problem consists of finding an $x \in \mathcal{X}$ such that $F(x) = 1$, in the least amount of queries to $F$ possible.

### Theorem

*Let $F : \mathcal{X} \to \{0,1\}$ be a function, $t = |\{x|F(x) = 1\}|$, and $N = |\mathcal{X}|$. Then, Grover's algorithm finds an $x$ such that $F(x) = 1$ with constant probability with $O\left(\sqrt{\frac{N}{t}}\right)$ queries to $F$. Moreover, this algorithm is optimal.*

# Our algorithm

## Algorithm for finding a $(r, k)$–SC

Input: $t \in \mathbb{N}$, $k' \in \mathbb{N}$.

1. Execute the $(r-1, k')$–SC algorithm $t$ times to find $t$ distinct $(r-1, k')$–SC in $h_1, \ldots, h_{k'}$.
   Let $T$ be the set of these $(r-1, k')$–SC.

# Our algorithm

## Algorithm for finding a $(r, k)$–SC

Input: $t \in \mathbb{N}$, $k' \in \mathbb{N}$.

1. Execute the $(r-1, k')$–SC algorithm $t$ times to find $t$ distinct $(r-1, k')$–SC in $h_1, \ldots, h_{k'}$.
   Let $T$ be the set of these $(r-1, k')$–SC.
2. Define $F : \mathcal{X} \to \{0, 1\}$ as follows:

$$F(x) = \begin{cases} 1, & \text{if there exists } (x_0, x_1, \ldots, x_{r-1}) \in T \text{ such that} \\ & \forall 1 \leq m \leq k - k', h_m(x) = h_{k'+m}(x_0), \\ 0, & \text{otherwise.} \end{cases}$$

# Our algorithm

## Algorithm for finding a $(r, k)$–SC

Input: $t \in \mathbb{N}$, $k' \in \mathbb{N}$.

1. Execute the $(r-1, k')$–SC algorithm $t$ times to find $t$ distinct $(r-1, k')$–SC in $h_1, \ldots, h_{k'}$.
   Let $T$ be the set of these $(r-1, k')$–SC.
2. Define $F : \mathcal{X} \to \{0, 1\}$ as follows:

$$F(x) = \begin{cases} 1, & \text{if there exists } (x_0, x_1, \ldots, x_{r-1}) \in T \text{ such that} \\ & \forall 1 \leq m \leq k - k', h_m(x) = h_{k'+m}(x_0), \\ 0, & \text{otherwise.} \end{cases}$$

3. Execute Grover's algorithm to find an $x$ such that $F(x) = 1$

# Our algorithm

## Algorithm for finding a $(r,k)$–SC

Input: $t \in \mathbb{N}$, $k' \in \mathbb{N}$.

1. Execute the $(r-1,k')$–SC algorithm $t$ times to find $t$ distinct $(r-1,k')$–SC in $h_1,\ldots,h_{k'}$.
   Let $T$ be the set of these $(r-1,k')$–SC.
2. Define $F : \mathcal{X} \to \{0,1\}$ as follows:

$$F(x) = \begin{cases} 1, & \text{if there exists } (x_0,x_1,\ldots,x_{r-1}) \in T \text{ such that} \\ & \forall 1 \le m \le k-k', h_m(x) = h_{k'+m}(x_0), \\ 0, & \text{otherwise.} \end{cases}$$

3. Execute Grover's algorithm to find an $x$ such that $F(x) = 1$
4. Find $(x_0,x_1,\ldots,x_{r-1})$ in $T$ and output $(x_0,x_1,\ldots,x_{r-1},x)$.

# Summary of our results

| Problem | Lower bound | Upper bound | Tight? |
|---|---|---|---|
| $k$-RSC | $\Omega\left(N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ | $O\left(N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ | Yes |
| $(1, k)$-SC | $\Omega\left(N^{k/5}\right)$ | $O\left(N^{k/4}\right)$ | No |
| $(r, k)$-SC | ? | $O\left(N^{k/(2+2r)}\right)$ | - |
| $(r, k)$-TSC | $\Omega\left(N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ | ? | - |
| $(r, k)$-ITSC | $\Omega\left(N^{\frac{2^k-1}{2^{k+1}-1}}\right)$ | ? | - |

- State of the art
- Our contribution
- Relationship between the problems

# Conclusion

## Open questions

- Can we close the gap for $k$-RSC when $k$ is not a constant ?

# Conclusion

## Open questions

- Can we close the gap for $k$-RSC when $k$ is not a constant ?
- Can we close the gap for $(1, k)$-SC ?

# Conclusion

## Open questions

- Can we close the gap for $k$-RSC when $k$ is not a constant ?
- Can we close the gap for $(1, k)$-SC ?
- Can we find a lower bound for $(r, k)$-SC when $r \geq 2$ ?

## Open questions

- Can we close the gap for $k$-RSC when $k$ is not a constant ?
- Can we close the gap for $(1, k)$-SC ?
- Can we find a lower bound for $(r, k)$-SC when $r \geq 2$ ?
- Can we have more precise analysis of the problems $(r, k)$-TSC and $(r, k)$-ITSC ?

# Conclusion

## Open questions

- Can we close the gap for $k$-RSC when $k$ is not a constant ?
- Can we close the gap for $(1, k)$-SC ?
- Can we find a lower bound for $(r, k)$-SC when $r \geq 2$ ?
- Can we have more precise analysis of the problems $(r, k)$-TSC and $(r, k)$-ITSC ?

Thank you for your attention!

Samuel Bouaziz-Ermann, Alex B. Grilo, and Damien Vergnaud.
Quantum security of subset cover problems.
Cryptology ePrint Archive, Paper 2022/1474, 2022.
https://eprint.iacr.org/2022/1474.

Qipeng Liu and Mark Zhandry.
On finding quantum multi-collisions.
In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 189–218. Springer, Heidelberg, May 2019.

Quan Yuan, Mehdi Tibouchi, and Masayuki Abe.
On subset-resilient hash function families.
*Designs, Codes and Cryptography*, 90, 03 2022.