Pseudorandom Quantum States with Proof of Destruction and applications

Amit Behera, Zvika Brakerski, Or Sattath, Omri Shmueli
BGU Weizmann Institute BGU Tel Aviv University



Qcrypt 2023

https://eprint.iacr.org/2023/543.pdf

Ben-Gurion University

This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No 756482)

European Research Council
Established by the European Commission

Pseudorandom quantum states

Ji-Liu-Song'18

Recall: Pseudorandom States definition

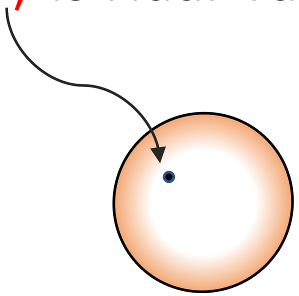
A quantum poly-time (QPT) algorithm G is a **pseudorandom state (PRS) generator** if

- given key $k \in \{0,1\}^\lambda$, $G(k)$ outputs n-qubit state $|\psi_k\rangle$
- for all t, for all poly-time algorithms D (called a **distinguisher**),

$|\psi_k\rangle = G(k)$ for random $k \in \{0,1\}^\lambda$

$$D\left(|\psi_k\rangle^{\otimes t}\right) \approx D\left(|\vartheta\rangle^{\otimes t}\right)$$

$|\vartheta\rangle$ is Haar-random



Recall: Pseudorandom States definition

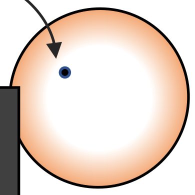
A quantum poly-time (QPT) algorithm G is a **pseudorandom state (PRS) generator** if

- given key $k \in \{0,1\}^{\lambda}$, $G(k)$ outputs n-qubit state $|\psi_k\rangle$
- for all t, for all poly-time algorithms D (called a **distinguisher**),

$|\psi_k\rangle = G(k)$ for random $k \in \{0,1\}^{\lambda}$

$$D\left(|\psi_k\rangle^{\otimes t}\right) \approx D\left(|\vartheta\rangle^{\otimes t}\right)$$

$|\vartheta\rangle$ is Haar-random



A PRS generator is different from a **state t-design**, where indistinguishability only holds for some *fixed* t.

Pseudorandom *function-like* states

A quantum poly-time algorithm G is a **PRFS generator** if

- given **key** $k \in \{0,1\}^{\lambda}$ and **input** $x \in \{0,1\}^{d}$, $G(k,x)$ outputs n-qubit state $|\psi_{k,x}\rangle$
- for all t, for all distinct inputs $x_1, \dots, x_s$, for all poly-time distinguishers D

$$D\left(|\psi_1\rangle^{\otimes t}, \dots, |\psi_s\rangle^{\otimes t}\right) \quad \approx \quad D\left(|\vartheta_1\rangle^{\otimes t}, \dots, |\vartheta_s\rangle^{\otimes t}\right)$$

$|\psi_i\rangle$'s sampled by:
- sampling random $k \in \{0,1\}^{\lambda}$
- setting $|\psi_i\rangle = G(k, x_i)$ for $i = 1, \dots, s$

$|\vartheta_i\rangle$'s sampled by:
- Independently sampling Haar-random $|\vartheta_i\rangle$ for $i = 1, \dots, s$

Important: the distinguisher D is allowed to depend on $x_1, \dots, x_s$!

Quantum States with Proof of destruction

Motivation

$|\sigma\rangle$

Intrinsic worth. Eg. money

How can Alice verify?

If σ is classical, no hope!

Can you destroy it?

$| \ \rangle$

σ

p

Bank

$p \leftarrow Destruct(|\sigma\rangle)$

✔ ✘ $\leftarrow Verify(k, p)$

$|\sigma\rangle \leftarrow Mint(k)$

Known constructions and comparison to this work

Reference	Based on	Pseudorandomness	Proof of destruction
BS’16, Wie’69, MVW’13, PYJ+’12 CLLZ’21, Shm’22	BB84/Subspace/Coset states	✗	✓

Known constructions and comparison to this work

Reference	Based on	Pseudorandomness	Proof of destruction
BS’16, Wie’69, MVW’13, PYJ+’12 CLLZ’21, Shm’22	BB84/Subspace/Coset states	✗	✓
JLS’21, BS’20	Random phase state	✓	✗

Known constructions and comparison to this work

Reference	Based on	Pseudorandomness	Proof of destruction
BS'16, Wie'69, MVW'13, PYJ+'12 CLLZ'21, Shm'22	BB84/Subspace/Coset states	✗	✓
JLS'21, BS'20	Random phase state	✓	✗
This work	Random phase state on a hidden set	✓	✓

Definitions

Pseudorandom States with proof of destruction (PRSPD)

Keyspace $\{0,1\}^\lambda$ associated with a triplet of efficient algorithms

- $|\psi_k\rangle \leftarrow Gen(k)$
- $p \leftarrow Destruct(|\psi_k\rangle)$
- ✔ ✘ $\leftarrow Ver_k(p)$

Correctness: $\Pr\left[k \leftarrow \{0,1\}^\lambda, |\psi_k\rangle \leftarrow Gen(k), p \leftarrow Destruct(|\psi_k\rangle) : 1 \leftarrow Ver_k(p)\right] = 1.$

Security

- Pseudorandomness
  - Same as the Pseudorandom States
- Unforgeability of proof of destruction

Unforgeability game

p_1

p_2

p_{t+1}

$Ver_k(p_i)$

QPT adversaries

Verification Oracle

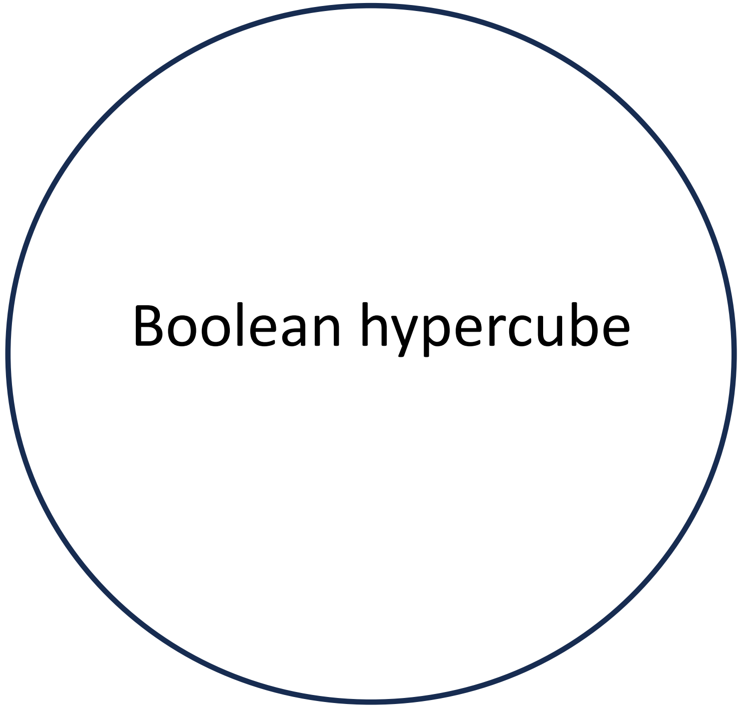
$|\psi_k\rangle^{\otimes t}$

Construction

Construction

Recall Ji-Liu-Song'19 (Simplified by Brakerski-Shmueli'20)

- Pseudorandom function family (PRF): $\{f_k\}_{k \in K}$
- Same keyspace K.



$$|\psi_k\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_x (-1)^{f_k(x)} |x\rangle$$

Random phase state

Sparsifying the construction

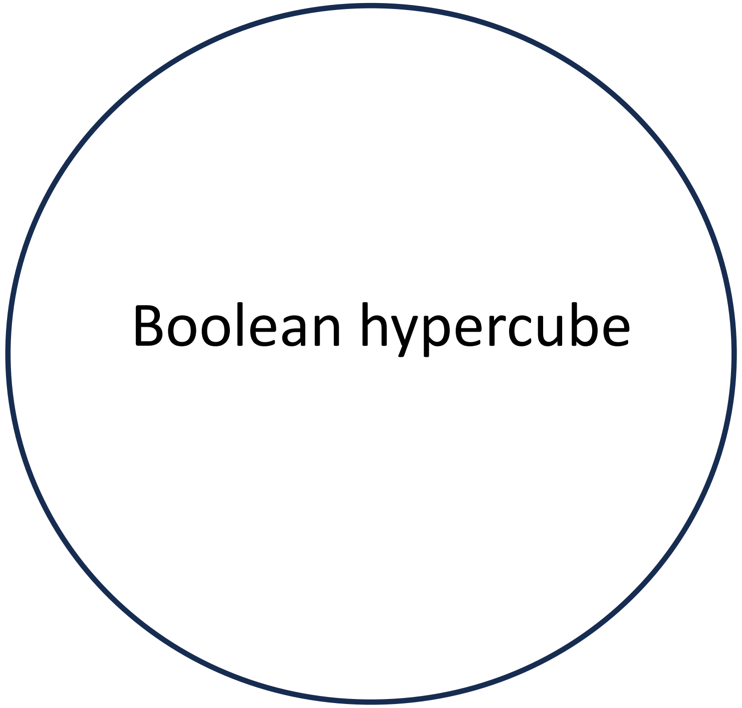
<u>PRS with support on all strings</u>

$$|\psi_k\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_{x} (-1)^{f_k(x)} |x\rangle$$

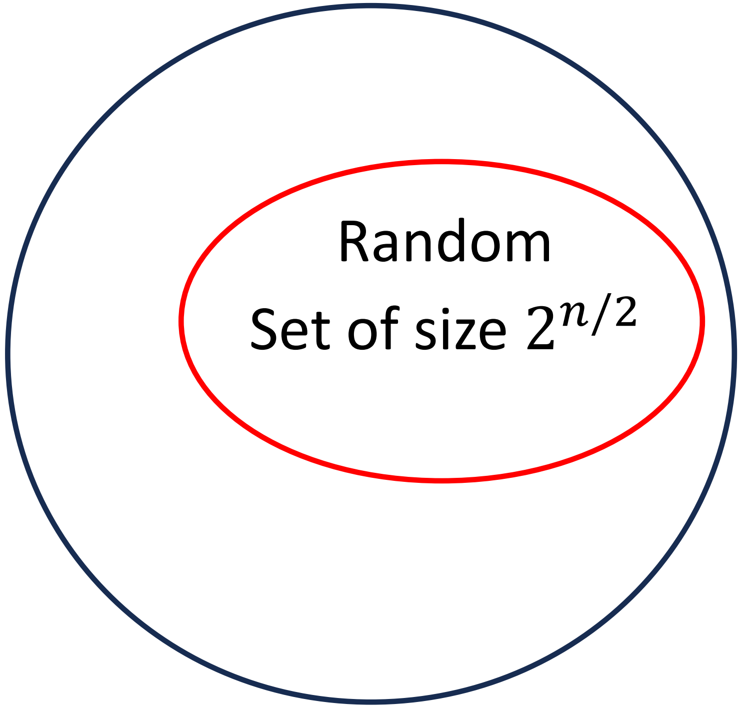
<u>PRSPD with support on a hidden set</u>

$$|\psi_{k,r}\rangle = \frac{1}{2^{\frac{n}{4}}} \sum_{x \in S_r} (-1)^{f_k(x)} |x\rangle$$

S_r : Pseudorandom set of size $2^{n/2}$



Random phase state



Sparsifying the construction

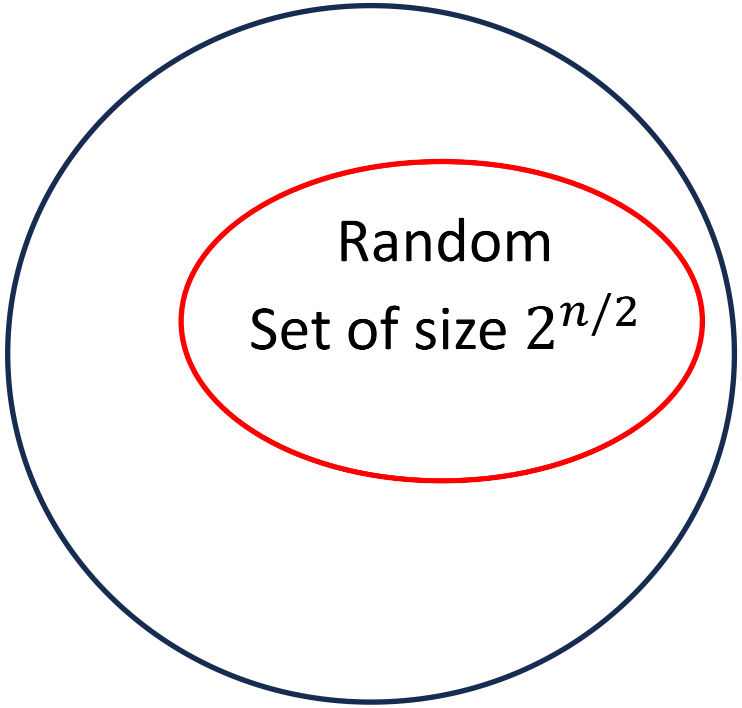
- PRF: $\{f_k\}_{k \in K}$
- Pseudorandom Permutation (PRP): $\{P_r\}_{r \in R}$
- Keyspace $K \times R$

PRSPD with support on a hidden set

$$|\psi_{k,r}\rangle = \frac{1}{2^{\frac{n}{4}}} \sum_{x \in S_r} (-1)^{f_k(x)} |x\rangle$$

$S_r = \{P_r(z) | z \in 0^{n/2} \times \{0,1\}^{\frac{n}{2}}\}$ $\quad r \leftarrow Uniform(R)$ $\quad S_r$: Pseudorandom set of size $2^{n/2}$

Random phase state



- Destruct: computational basis measurement
- $Ver_{k,r}()$: Membership in S_r

Technical lemma for security proofs

$k \leftarrow \{0,1\}^{\lambda}$

$|\psi_k\rangle \leftarrow Gen(k)$

$\vec{x} = x_1, x_2, \ldots x_t \leftarrow \{0,1\}^n$

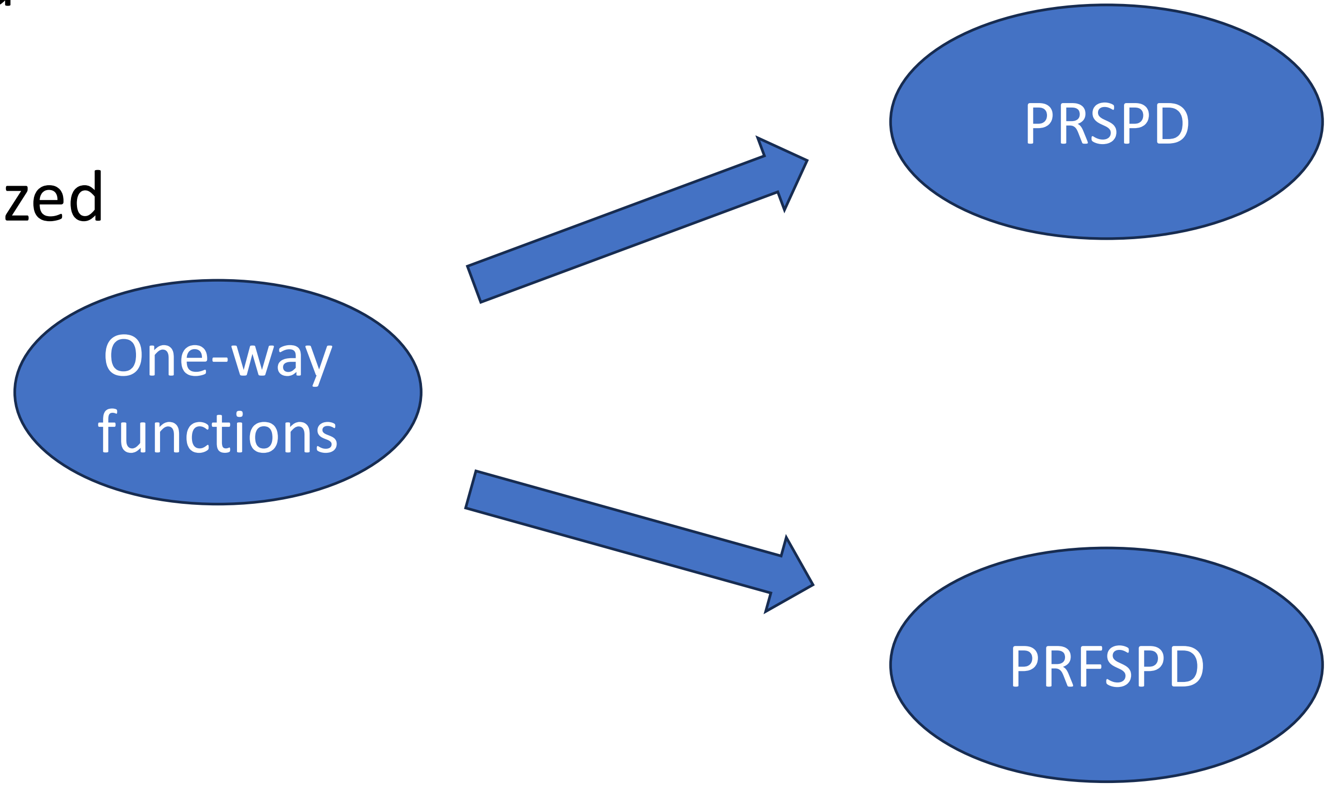
$|Sym_{\vec{x}}\rangle \propto \sum_{\pi} |x_{\pi(1)}, x_{\pi(2)}, \ldots x_{\pi(t)}\rangle$

$Ver_k(\cdot)$ $\approx$ $1_{\{x_1, x_2, \ldots, x_t\}}(\cdot)$

$|\psi_k\rangle^{\otimes t}$ $|Sym_{\vec{x}}\rangle$

Pseudorandom Function-like States with Proof of Destruction (PRFSPD)

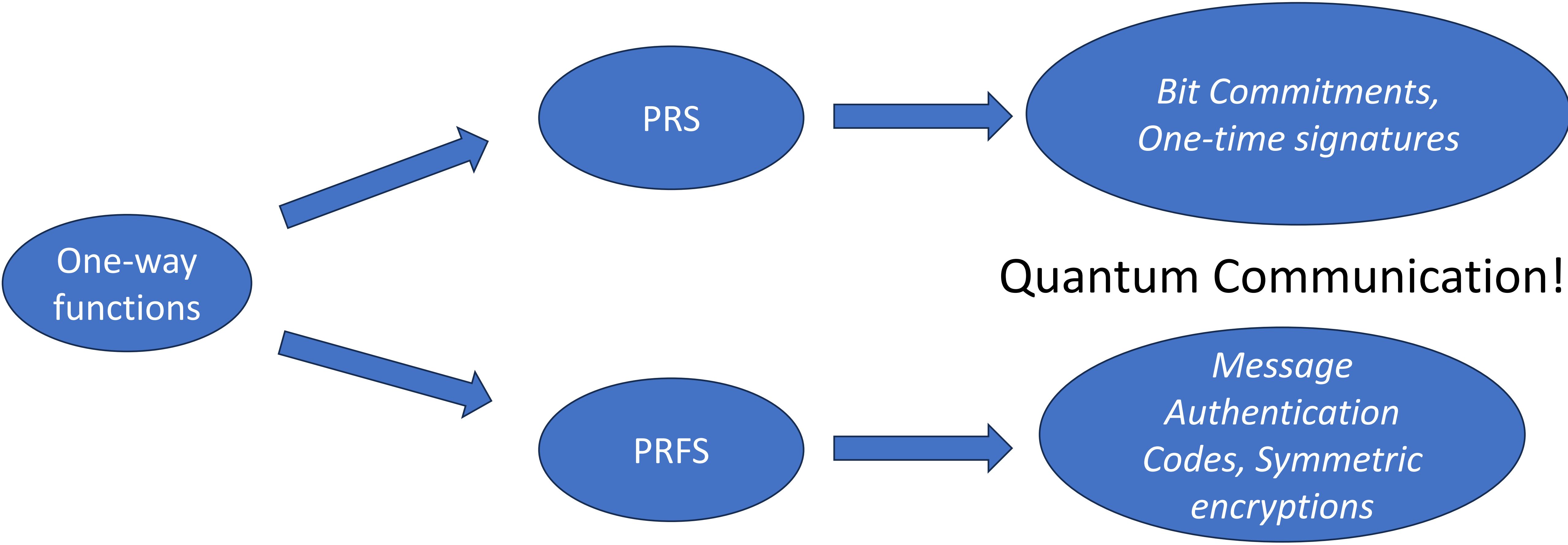
- Definition can be generalized
- Construction can be generalized



Applications

Applications of PRS, PRFS

(Ananth-Qian-Yuen'21, Morimae-Yamakawa'21, etc)

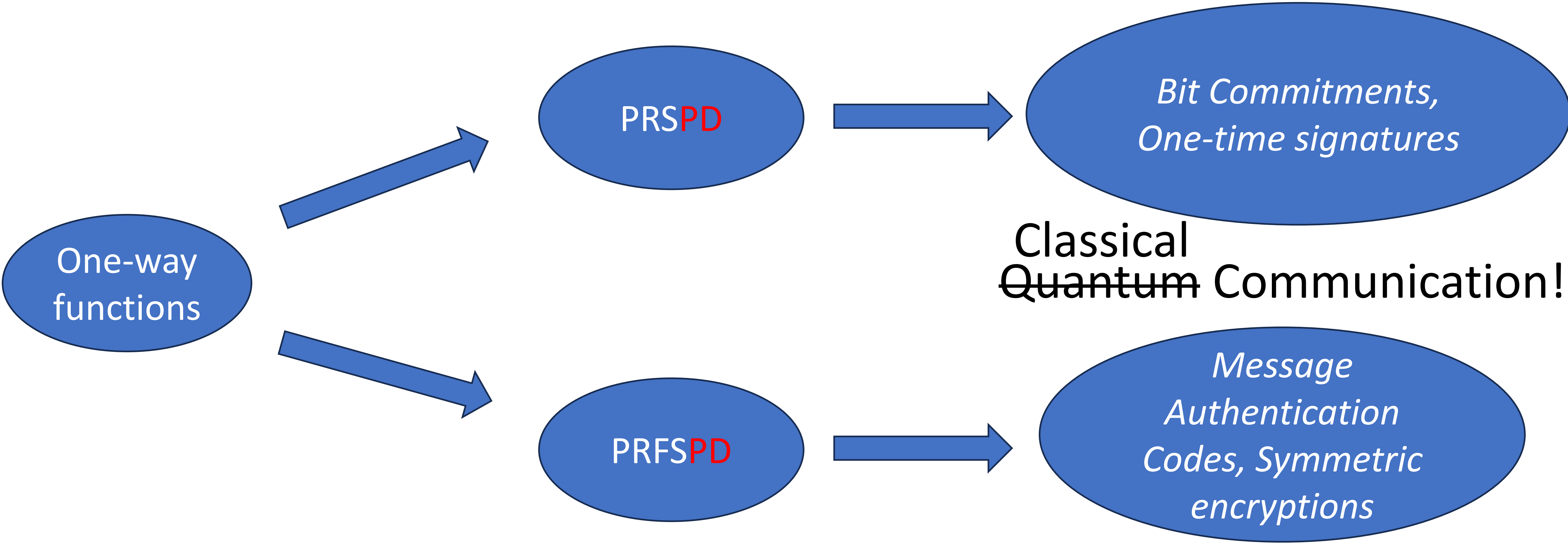


Black box

Necessary?

Applications of PRSPD, PRFSPD

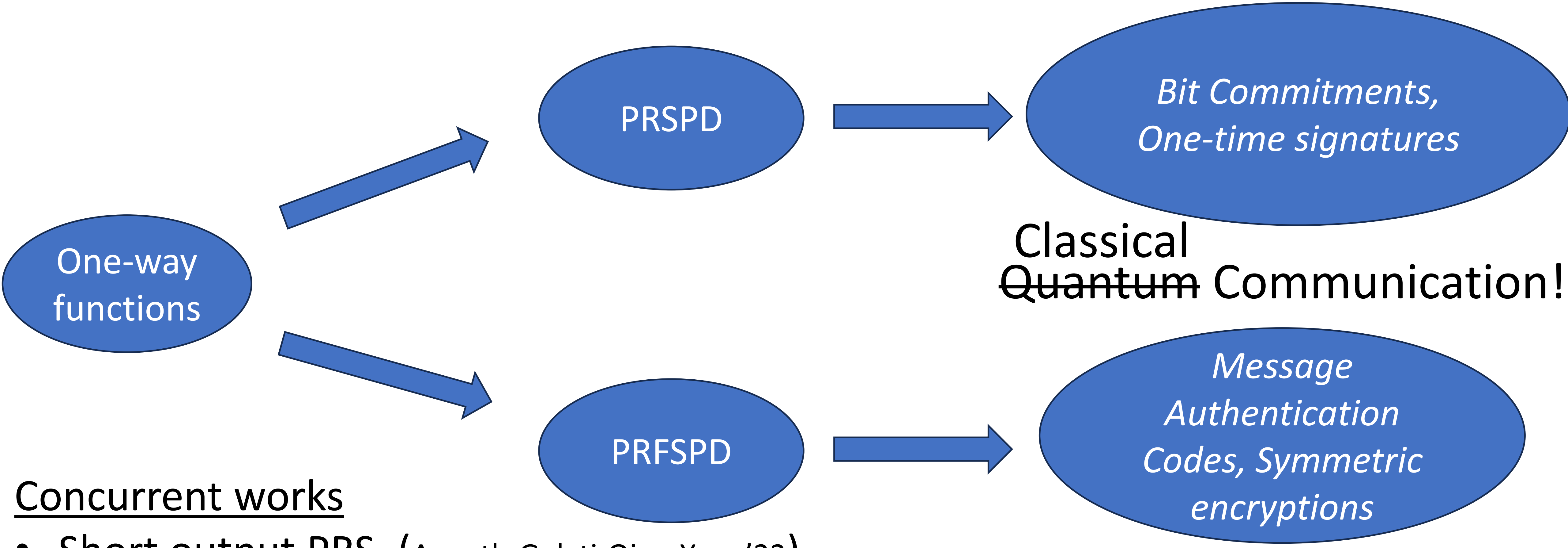
(Our Work)



Black box

Applications of PRSPD, PRFSPD

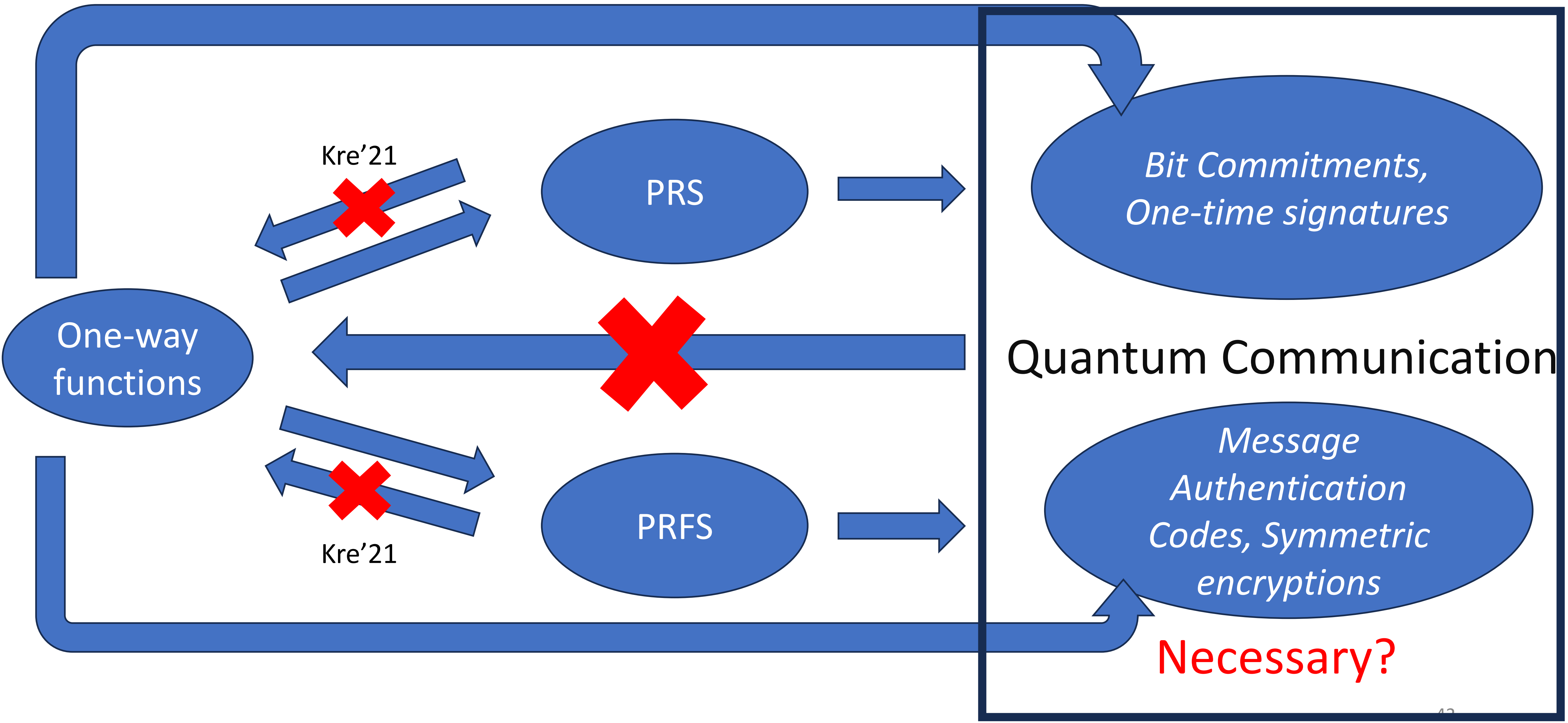
(Our Work)



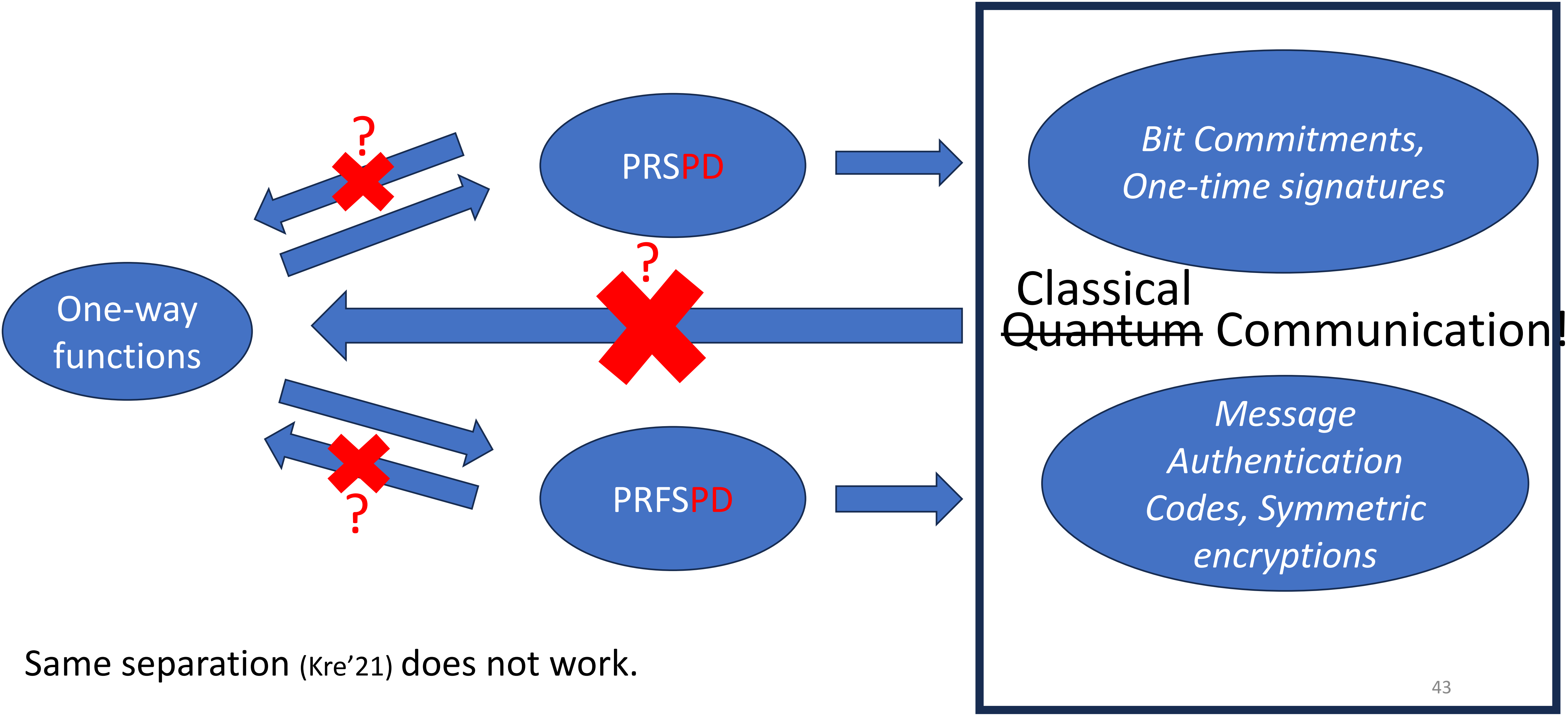
Concurrent works

- Short output PRS. (Ananth-Gulati-Qian-Yuen'22)
- Pseudo-deterministic PRG. (Ananth-Lin-Yuen'23)

Why do we care?

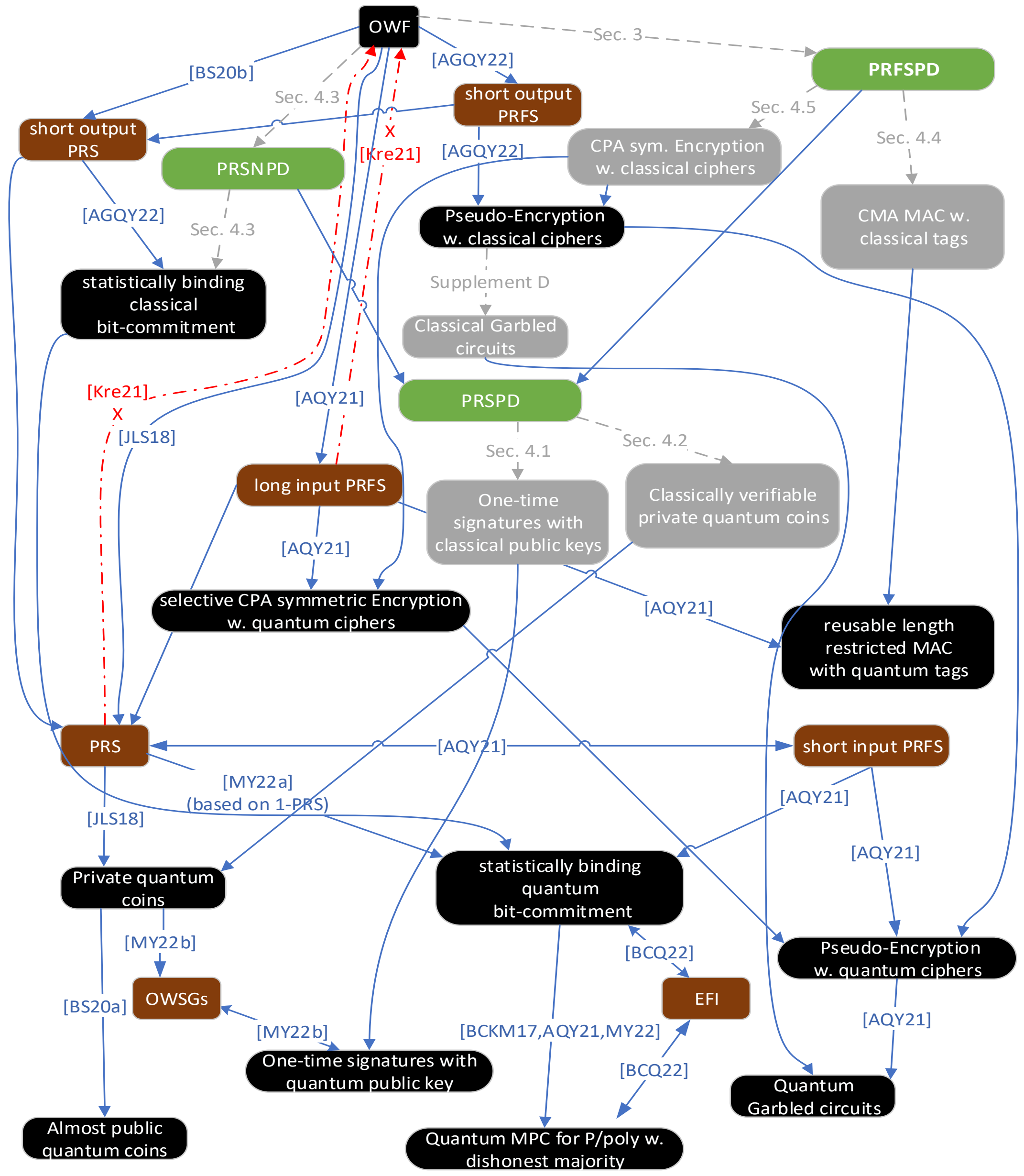


Why do we care?



Same separation (Kre'21) does not work.

Full picture currently

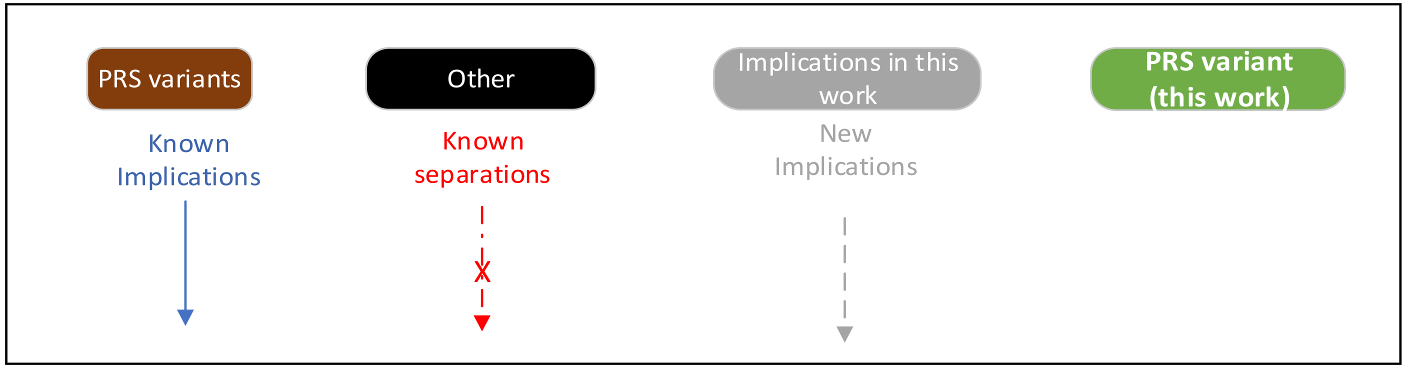


Welcome to the Jungle!

https://sattath.github.io/qcrypto-graph/

Simplify?

Legend



Open-problem

PRS (or PRFS) $\not\Rightarrow$ One-way functions (Kre'21).

PRSPD (or PRFSPD) $\overset{?}{\not\Rightarrow}$ One-way functions.

Separation $\Longrightarrow$ Most classical minicrypt primitives do not need one-way functions or quantum communication.

Related question: Separation of short-output PRS from OWF?

Template for the applications

Template for dequantazing PRS/PRFS applications

Quantum communication

k

PRS/PRFS state

$m, |\psi_{k,m}\rangle$

k

$MAC.Verify_k\,(m)$

$|\psi\rangle \leftarrow PRFS.Gen\,(k, m)$

✔ ❌ $\leftarrow SWAP(|\psi\rangle, |\psi_{k,m}\rangle)$

$MAC.Sign(k, m)$

$|\psi_{k,m}\rangle \leftarrow PRFS.Gen(k, m)$

Template for dequantazing PRS/PRFS applications

Classical

- ~~Quantum~~ communication

k

Proof of destruction

of the PRS/PRFS state

m, p

k

$MAC.Verify_k\,(p, m)$

✔ ✘ $\leftarrow PRFS.Verify_k\,(p, m)$

$MAC.Sign(k, m)$

$|\psi_{k,m}\rangle \leftarrow PRFS.Gen(k, m)$

$p \leftarrow PRFS.Destruct\,(k, m)$

Works in most but not in all cases!

Thank you!

Challenges in this template

One-way functions → One-time signatures (Lam79)

y

k'

$OWF(k') = y?$

$k \leftarrow Uniform$

$y \leftarrow \boxed{\text{OWF}} \leftarrow k$

Public key of the signature are OWF images (like y)

Challenges in this template

PRS ➡ One-time signatures (MY22)

$|\psi_k\rangle$

k'

$Gen(k') \approx |\psi_k\rangle$?

$k \leftarrow Uniform$

$|\psi_k\rangle \leftarrow \boxed{\text{Gen}} \leftarrow k$

Public key of the signature are PRS states (like $|\psi_k\rangle$)

Challenges in this template

PRSPD ⇒ One-time signatures (Morimae-Yamakawa'22)

p_k

k'

$Ver_{k'}(p_k)$=1?

$k \leftarrow Uniform$

$p_k \leftarrow$ Destruct·Gen $\leftarrow k$

- Pseudorandomness/unforgeability is not enough!
- Add a dummy key $\tilde{k}$ that accepts Ver on all proofs.
  - All previous security guarantees hold!
  - Adversary can output $\tilde{k}$ trivially.

Proofs of destruction are not one-way!

Solution: Change the verification algorithm to rule out dummy keys!

Thank you!

Example of an application: MAC construction

Hurdles in finding a separation

Other studied variants of Pseudorandom states

- PRFS (potentially stronger than PRS)
- Short Output PRS (potentially stronger)
- EFI (potentially weaker)
- One-way states (potentially weaker)