One-Shot Signatures and Applications

Aggelos Kiayias
University of Edinburgh and IOG
Joint work with Ryan Amos, Marios Georgiou, Mark Zhandry
Slides credits: Marios Georgiou & AK

Classical
Quantum

One-Shot **QSig**

1. Quantum Retrieval Games
2. Tokenized Signatures
3. One-Shot Signatures

- sk: quantum **signing** key
- vk: classical **verification** key

Common syntax

- Gen(1^n): (sk, vk)
- Sign(sk, m): σ
- Ver(vk, m, σ): b

Correctness

If (sk, vk) ← Gen(1^n) then Ver(vk, m, Sign(sk, m)) = 1 for any message m.

Security

High level: sk can sign only a single message. It collapses.

Classical
Quantum

Quantum Retrieval Games

A C

sk

$\longleftarrow$ $(sk, vk) \leftarrow \mathrm{Gen}(1^n)$

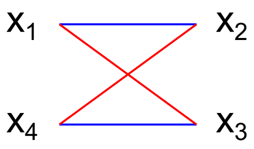
$\longrightarrow$

σ_0, σ_1

A wins if:

- $\mathrm{Ver}(vk, 0, \sigma_0) = 1$
- $\mathrm{Ver}(vk, 1, \sigma_1) = 1$

Construction: Hidden Matching Problem [Gavinsky 2012]
Security: Unconditional



Tokenized Signatures

Classical
Quantum

A

C

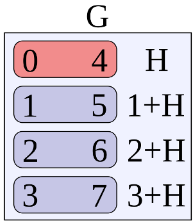
sk,vk

(sk, vk) ← Gen(1^n)

σ_0,σ_1

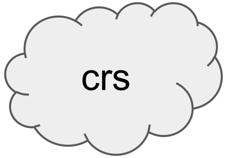
A wins if:

- Ver(vk, 0, σ_0) = 1
- Ver(vk, 1, σ_1) = 1



Construction: Hidden Cosets [CLLZ21]
Security: iO + OWF

One-Shot Signatures



Classical
Quantum

A

C

sk,vk

(sk, vk) ← Gen(1^n)

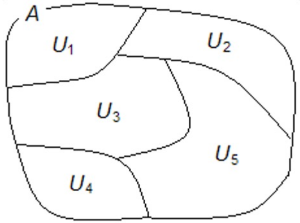
σ_0, σ_1, vk

A wins if:

- Ver(vk, 0, σ_0) = 1
- Ver(vk, 1, σ_1) = 1

Constructions:

- Partition Z_2^n into subsets and access partition via
  - Quantum oracles
  - Classical oracles
- Plain Model: Candidate constructions by obfuscating the oracles



One-Shot Signatures from One-Shot Chameleon

Classical
Quantum

Syntax

Signing key

Verification key

Message

Signature

- Gen(1^n): (sk, y)
- Invert(sk, x): r
- H(x, r): y

Reduction
If one can sign both 0 and 1,
it can find a collision.

Correctness

If (sk, y) ← Gen(1^n) then H(x, r) = y for any input x, where r ← Invert(sk, x).

Security

H is collision resistant.

Classical
Quantum

Equivocal Hash Functions

Syntax

- Gen(1^n): (sk, y, Q(.))
- Equiv(sk, b): x
- H(x): y

Correctness/Equivocality

If (sk, y) $\leftarrow$ Gen(1^n) then H(x) = y and Q(x)=b for any input x, where x $\leftarrow$ Invert(sk, b).

Security

H is collision resistant.

Classical
Quantum

OS Chameleon and Equivocal Hash Functions

Theorem. Equivocal hash functions ⇔ One-shot Chameleon

- Gen(1^n): (sk, y, Q(.))
- Equiv(sk, b): x
- H(x): y

- Gen’(1^n): (sk, y’)
- Invert(sk, x’): r’
- H’(x’, r’): y

x’ is in {0,1}

Proof “<=” we set Q to be the first bit. Define H(x) = H’(x_0, $x_1 \ldots x_{k-1}$).

For Equiv, given *sk*,*b*, run Invert(*sk*,*b*) to obtain r’ such that H’(b,r’)=y, it follows that b||r’ satisfies the Q predicate and is a preimage.

“=>”Define H’(x’,r’) = H(r’) || Q(r’) XOR x’ and y’ = y||0
For invert(sk, x’) : run Equiv(sk, x’) to obtain r’ such that Q(r’)=x’ and H(r’)=y.
Observe that H’(x’, r’) = H(r’) || Q(r’) XOR x’ = y || x’ XOR x’ = y || 0 = y’

The quantum flavours of collision resistance

▨ Classical
▨ Quantum

- Collision resistant.
  - Hard to find distinct x0, x1 such that H(x0)=H(x1)
- Unequivocal
  - No efficient adversary can find a hash *y*, and predicate Q such that later given b, it is possible to find a pre-image x with Q(x)=b and H(x)=y
- Collapsing
  - Let A(y) be the preimage set of any y. Having access to a superposition of A(y) is no more useful than having a random element of A(y).

Observe: Collapsing => Unequivocal => Collision resistant

Hash functions with quantum capabilities

- Initially non-collapsing introduced as a problem.
  - Observe that it is feasible to be CR and non-collapsing: due to no-cloning, despite the uncertainty in a pre-image set A(y) this state cannot be cloned and measured twice to break collision resistance.
  - **What is the potential problem**: non-committing hashing - [U16]
  - *However it can also be a good thing*
- (Classically) Collision Resistant & non-collapsing
  - **Application:** quantum lightning [Z17] (a stronger version of quantum money)
    - Gen => bolt
    - Verify(bolt) => bolt, serial number.
    - Not possible to create two distinct bolts with the same serial.
- (Classically) Collision resistant & equivocal
  - More applications! Quantum lightning => decentralized cryptocurrency, but one-shot signatures can do more: e.g., decentralized smart contracts without PoW/erasures/VDFs

Constructing One-Shot Signatures

▨ Classical
▨ Quantum

Approach via one-shot Chameleon for one-bit messages

- Gen(1^n): (sk, y)
- Invert(sk, *b*): r
- H(*b*, r): y

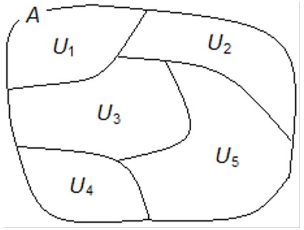
Key question: how do we implement Invert

Pick-one trick from [ARU14]. Grover's search is adapted appropriately.

Suppose A(y) the preimage set of y

- Superposition of A(y); => elements of the form (0,r) and (1,r)
- Apply phase shift |(-1)^H(*b*,r) (*b*,r)> and diffusion **I** -2*| A(y) >< A(y)|
- Measure & repeat until a suitable solution is produced.
  - (note: for efficiency, there should be sufficiently many *r* choices |A(y)| / |A(y) : H(<u>*b*</u>,r)=y| is polynomial)

One-Shot Chameleon From Oracles



Classical
Quantum

Quantum Oracle

- Partition $\{0,1\}^n$ into $2^{n/2}$ sets $\{U_y\}$ of size $2^{n/2}$
- Oracles
  1. $H(x) = y$ if $x \in U_y$
  2. Reflect(state,y) =
     - If state = $|y, U_y\rangle$ return $-|y, U_y\rangle$
     - If state = $|y, \perp U_y\rangle$ return $|y, \perp U_y\rangle$
- Gen:
  1. Evaluate H on uniform superposition
  2. Measure output register to get (sk, y)
     - Input register collapses to uniform superposition of y's preimages
- Invert(sk, *b*):
  1. Run Grover's search using Reflect.
  2. Retrieve uniform superposition of preimages starting with *b*.
  3. Measure.

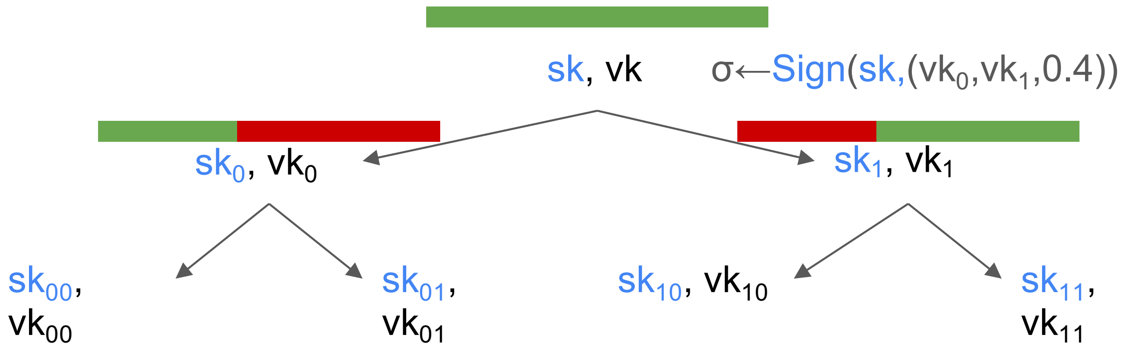
Classical Oracle

- Partition $\{0,1\}^n$ into $2^{n/2}$ cosets $\{U_i\}$ of size $2^{n/2}$
- Oracles
  1. $H(x) = y$ if $x \in U_y$
  2. **$H^\perp(x,y)$ =**
     - **If $x \in U_y^\perp$, accept**
     - **If $x \notin U_y^\perp$, reject**
- Gen:
  1. Evaluate H on uniform superposition
  2. Measure output register to get (sk, y)
     - Input register collapses to uniform superposition of y's preimages
- Invert(sk, b):
  1. Run Grover's search using **QFT·$H^\perp(\cdot,y)$ QFT**.
  2. Retrieve uniform superposition of preimages starting with *b*.
  3. Measure.

Applications

Applications: Budget Signatures

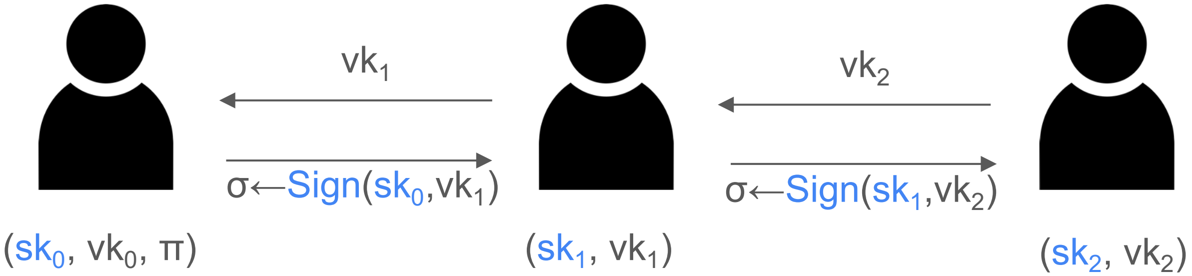
Classical
Quantum



PoW Coins with Classical Communication

Classical
Quantum

- To mine a new coin:
  - $(sk_0, vk_0) \leftarrow$ Gen
  - Run a proof of work on vk_0 to generate proof π.
  - (sk_0, vk_0, π) is the coin
- To send the coin:
  - Receiver generates new $(sk_1, vk_1) \leftarrow$ Gen and sends vk_1 to the sender.
  - Sender signs vk_1: $\sigma \leftarrow \mathrm{Sign}(sk_0, vk_1)$ and sends (vk_0, π, σ) to the receiver.



Pow Coins: Improvements

▨ Classical
▨ Quantum

- Succinctness:
  - Compress $(vk_0, \pi), (vk_1, \sigma_1), \ldots, (vk_n, \sigma_n)$ into succinct proof
- Protecting privacy:
  - Use Zero-Knowledge, (as above)

Ordered Signatures

▨ Classical
▨ Quantum

- Each signature is associated with a time tag
- **Security**: One cannot sign a message with a "past" tag

$(vk_0,\sigma_0) \longrightarrow (vk_1,\sigma_1) \longrightarrow (vk_2,\sigma_2)$

$\sigma_0 \leftarrow \mathrm{Sign}(sk_0,(vk_1,m_1,t_1))$ $\quad$ $\sigma_1 \leftarrow \mathrm{Sign}(sk_1,(vk_2,m_2,t_2))$

- Verification:
  1. Verify all signatures
  2. Verify that $t_i > t_{i-1}$
- Proof of burn by signing at $t = \infty$

Public-coin Proofs of Quantumness

Classical
Quantum

OSS => PPQ

crs

How can a prover convince a verifier he is quantum

P		V
(sk, vk) ← Gen	$\xrightarrow{\text{vk}}$	
	$\xleftarrow{\text{m}}$	$m \leftarrow \{0,1\}^n$
$\sigma \leftarrow$ Sign(sk,m)	$\xrightarrow{\sigma}$	
		Ver(vk,m,σ) = 1

If a classical P is convincing then by rewinding we could get signatures of two messages => breaks one-shot

Classical
Quantum

One-Shot Signatures with Trapdoors

crs(tr)

A

C

sk,vk

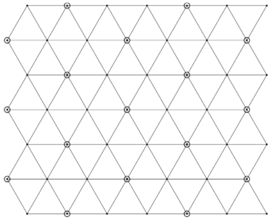
(sk, vk) ← Gen(1^n)

σ_0, σ_1, vk

A wins if:

- Ver(**tr**, vk, 0, σ_0) = 1
- Ver(**tr**, vk, 1, σ_1) = 1

Construction: [BCMVV18]
Assumption: Learning with Errors



Classical
Quantum

Summary **QSig**

	Honest Key Generation	**Dishonest Key Generation**
Private Verification	Quantum Retrieval Games (unconditional)	Trapdoor One-Shot Signatures (LWE)
Public Verification	Tokenized Signatures (iO + OWF)	One-Shot Signatures (only w.r.t. an oracle)

▨ Classical
▨ Quantum

Directions

1. Constructing Equivocal hash functions
   a. E.g., via light weight hash function approach as in [Z19], other techniques?
2. OSS design approaches and security
   a. Lower bounds for finding collisions against the partition oracle H, given $H^{\perp}(x,y)$
   b. We based the construction on one-shot Chameleon; other design techniques?
3. OSS without oracles
   a. E.g., from iO, similarly to tokenized signatures.
      Apply obfuscation to the classical oracle construction from one-shot Chameleon, is the resulting non-oracle construction secure?
4. Succinct and ZK proofs for chains of signatures for OSS applications.