# Developing Innovative Frameworks for Efficient Code-based Signatures

Edoardo Persichetti

9 June 2022

Charles E. Schmidt College of Science
Florida Atlantic University

# IN THIS TALK

- Introduction
- Traditional Approach
- Zero-Knowledge Protocols
- New Frameworks
- Conclusions

# Part I

## Introduction

# Post-Quantum Cryptography

In a few years time large-scale quantum computers might be reality.

In a few years time large-scale quantum computers might be reality.
But then (Shor, '95):

# POST-QUANTUM CRYPTOGRAPHY

In a few years time large-scale quantum computers might be reality.
But then (Shor, '95):

- RSA
- DSA
- ECC
- Diffie-Hellman key exchange
- and many others ... $\boxed{\textbf{not secure}}$ !

# Post-Quantum Cryptography

In a few years time large-scale quantum computers might be reality.
But then (Shor, '95):

- RSA
- DSA
- ECC
- Diffie-Hellman key exchange
- and many others ... $\boxed{\textbf{not secure}}$ !

$\rightarrow$ NIST's Post-Quantum Cryptography Standardization Call

# Post-Quantum Cryptography

In a few years time large-scale quantum computers might be reality.
But then (Shor, '95):

- RSA
- DSA
- ECC
- Diffie-Hellman key exchange
- and many others ... **not secure** !

$\rightarrow$ NIST's Post-Quantum Cryptography Standardization Call

Main areas of research:

- Lattice-based cryptography.
- Hash-based cryptography.
- Code-based cryptography (McEliece, Niederreiter).
- Multivariate cryptography.
- Isogeny-based cryptography.

# Motivation

Code-based cryptography has been doing really well for encryption/key establishment.

# Motivation

Code-based cryptography has been doing really well for encryption/key establishment.

3 finalists in NIST's process:

- Classic McEliece (binary Goppa)
- BIKE (QC-MDPC)
- HQC (QC Random Codes)

# MOTIVATION

Code-based cryptography has been doing really well for encryption/key establishment.

3 finalists in NIST's process:

- Classic McEliece (binary Goppa)
- BIKE (QC-MDPC)
- HQC (QC Random Codes)

The same cannot be said for code-based signatures.

# Motivation

Code-based cryptography has been doing really well for encryption/key establishment.

3 finalists in NIST's process:

- Classic McEliece (binary Goppa)
- BIKE (QC-MDPC)
- HQC (QC Random Codes)

The same cannot be said for code-based signatures.

Only 4 NIST submissions, all either broken or withdrawn.

# MOTIVATION

Code-based cryptography has been doing really well for encryption/key establishment.

3 finalists in NIST's process:

- Classic McEliece (binary Goppa)
- BIKE (QC-MDPC)
- HQC (QC Random Codes)

The same cannot be said for code-based signatures.

Only 4 NIST submissions, all either broken or withdrawn.

Yet, signature schemes are a crucial component in cryptography.

# MOTIVATION

Code-based cryptography has been doing really well for encryption/key establishment.

3 finalists in NIST's process:

- Classic McEliece (binary Goppa)
- BIKE (QC-MDPC)
- HQC (QC Random Codes)

The same cannot be said for code-based signatures.

Only 4 NIST submissions, all either broken or withdrawn.

Yet, signature schemes are a crucial component in cryptography.

Can we fix this?

### $[n, k]$ linear code over $\mathbb{F}_q$

A subspace of dimension $k$ of $\mathbb{F}_q^n$. Value $n$ is called length.

# Error-Correcting Codes

## $[n, k]$ linear code over $\mathbb{F}_q$

A subspace of dimension $k$ of $\mathbb{F}_q^n$. Value $n$ is called length.

## Hamming metric

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.

Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

# Error-Correcting Codes

## $[n, k]$ linear code over $\mathbb{F}_q$

A subspace of dimension $k$ of $\mathbb{F}_q^n$. Value $n$ is called length.

## Hamming metric

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.

Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

## Generator matrix

$G \in \mathbb{F}_q^{k \times n}$ defines the code as : $x \in \mathcal{C} \iff x = uG$ for $u \in \mathbb{F}_q^k$.
Not unique: $SG, S \in GL(k, q)$; Systematic form: $(I_k | M)$.

### $[n, k]$ LINEAR CODE OVER $\mathbb{F}_q$

A subspace of dimension $k$ of $\mathbb{F}_q^n$. Value $n$ is called length.

### HAMMING METRIC

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.
Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

### GENERATOR MATRIX

$G \in \mathbb{F}_q^{k \times n}$ defines the code as : $x \in \mathcal{C} \iff x = uG$ for $u \in \mathbb{F}_q^k$.
Not unique: $SG, S \in GL(k, q)$; Systematic form: $(I_k|M)$.

### PARITY-CHECK MATRIX

$H \in \mathbb{F}_q^{(n-k) \times n}$ defines the code as: $x \in \mathcal{C} \iff Hx^T = 0$ (syndrome).
Not unique: $SH, S \in GL(n - k, q)$; Systematic form: $(M^T|I_{n-k})$.

# Error-Correcting Codes

### $[n, k]$ linear code over $\mathbb{F}_q$

A subspace of dimension $k$ of $\mathbb{F}_q^n$. Value $n$ is called length.

### Hamming metric

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.
Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

### Generator matrix

$G \in \mathbb{F}_q^{k \times n}$ defines the code as : $x \in \mathcal{C} \iff x = uG$ for $u \in \mathbb{F}_q^k$.
Not unique: $SG, S \in GL(k, q)$; Systematic form: $(I_k | M)$.

### Parity-check matrix

$H \in \mathbb{F}_q^{(n-k) \times n}$ defines the code as: $x \in \mathcal{C} \iff Hx^T = 0$ (syndrome).
Not unique: $SH, S \in GL(n-k, q)$; Systematic form: $(M^T | I_{n-k})$.

$w$-error correcting: $\exists$ algorithm that corrects up to $w$ errors.

# Decoding Problems

In general, it is hard to decode random codes.

In general, it is hard to decode random codes.

### PROBLEM (GENERAL DECODING)

*Given*: $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
*Goal*: *find a word* $e \in \mathbb{F}_q^n$ *with* $\mathrm{wt}(e) \leq w$ *such that* $y - e = x \in \mathcal{C}_G$.

# Decoding Problems

In general, it is hard to decode random codes.

### Problem (General Decoding)

*Given:* $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e = x \in \mathcal{C}_G$.

Easy to see this is equivalent to the following.

# Decoding Problems

In general, it is hard to decode random codes.

**Problem (General Decoding)**

*Given:* $G \in \mathbb{F}_q^{k\times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e = x \in \mathcal{C}_G$.

Easy to see this is equivalent to the following.

**Problem (Syndrome Decoding)**

*Given:* $H \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

# Decoding Problems

In general, it is hard to decode random codes.

**Problem (General Decoding)**

*Given:* $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e = x \in \mathcal{C}_G$.

Easy to see this is equivalent to the following.

**Problem (Syndrome Decoding)**

*Given:* $H \in \mathbb{F}_q^{(n-k) \times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

NP-Complete (Berlekamp, McEliece and Van Tilborg, 1978; Barg, 1994).

# Decoding Problems

In general, it is hard to decode random codes.

**Problem (General Decoding)**

*Given:* $G \in \mathbb{F}_q^{k\times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e = x \in \mathcal{C}_G$.

Easy to see this is equivalent to the following.

**Problem (Syndrome Decoding)**

*Given:* $H \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

NP-Complete (Berlekamp, McEliece and Van Tilborg, 1978; Barg, 1994).

Unique solution when $w$ is below a certain threshold.

# Decoding Problems

In general, it is hard to decode random codes.

**Problem (General Decoding)**

*Given:* $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e = x \in \mathcal{C}_G$.

Easy to see this is equivalent to the following.

**Problem (Syndrome Decoding)**

*Given:* $H \in \mathbb{F}_q^{(n-k) \times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
*Goal: find a word* $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

NP-Complete (Berlekamp, McEliece and Van Tilborg, 1978; Barg, 1994).

Unique solution when $w$ is below a certain threshold.

Very well-studied, solid security understanding (ISD).

# How to Do Code-Based Cryptography?

Use hard problems from coding theory.

# How to Do Code-Based Cryptography?

Use hard problems from coding theory.

To get trapdoor, need one more ingredient.

# How to Do Code-Based Cryptography?

Use hard problems from coding theory.

To get trapdoor, need one more ingredient.

**Assumption (Code Indistinguishability)**

*Let $M$ be a matrix defining a code. Then $M$ is indistinguishable from a randomly generated matrix of the same size.*

# How to Do Code-Based Cryptography?

Use hard problems from coding theory.

To get trapdoor, need one more ingredient.

**Assumption (Code Indistinguishability)**

*Let M be a matrix defining a code. Then M is indistinguishable from a randomly generated matrix of the same size.*

Hardness of assumption depends on chosen code family.

# How to Do Code-Based Cryptography?

Use hard problems from coding theory.

To get trapdoor, need one more ingredient.

**Assumption (Code Indistinguishability)**

*Let $M$ be a matrix defining a code. Then $M$ is indistinguishable from a randomly generated matrix of the same size.*

Hardness of assumption depends on chosen code family.

Choose a code family with efficient decoding algorithm associated to description $\Delta$ and hide the structure.

# Part II

## Traditional Approach

# Intuition

Use the traditional SDP-based trapdoor.

# Intuition

Use the traditional SDP-based trapdoor.

Use hash-and-sign framework as in e.g. Full Domain Hash (RSA).

# INTUITION

Use the traditional SDP-based trapdoor.

Use hash-and-sign framework as in e.g. Full Domain Hash (RSA).

Given message *msg*, trapdoor OW function $f$ and hash function $\mathbf{H}$.

# INTUITION

Use the traditional SDP-based trapdoor.

Use hash-and-sign framework as in e.g. Full Domain Hash (RSA).

Given message *msg*, trapdoor OW function $f$ and hash function **H**.

Create signature $\sigma = f^{-1}(\mathbf{H}(msg))$.

# Intuition

Use the traditional SDP-based trapdoor.

Use hash-and-sign framework as in e.g. Full Domain Hash (RSA).

Given message *msg*, trapdoor OW function $f$ and hash function **H**.

Create signature $\sigma = f^{-1}(\mathbf{H}(msg))$.

Verify signature if $f(\sigma) = \mathbf{H}(msg)$.

# INTUITION

Use the traditional SDP-based trapdoor.

Use hash-and-sign framework as in e.g. Full Domain Hash (RSA).

Given message *msg*, trapdoor OW function $f$ and hash function $\mathbf{H}$.

Create signature $\sigma = f^{-1}(\mathbf{H}(msg))$.

Verify signature if $f(\sigma) = \mathbf{H}(msg)$.

For code-based, trapdoor is decoding: CFS scheme.
(Courtois, Finiasz, Sendrier, 2001)

# The CFS Scheme

Select hash function $\mathbf{H} : \{0,1\}^* \to \mathbb{F}_q^{(n-k)}$.

# The CFS Scheme

Select hash function $\mathbf{H}: \{0,1\}^* \to \mathbb{F}_q^{(n-k)}$.

## Key Generation

- Choose a code $\mathcal{C}$ (e.g. Goppa).
- SK: code description $\Delta$ for $\mathcal{C}$.
- PK: parity-check matrix $H$ in systematic form for $\mathcal{C}$.

# The CFS Scheme

Select hash function $\mathbf{H} : \{0,1\}^* \to \mathbb{F}_q^{(n-k)}$.

## Key Generation

- Choose a code $\mathcal{C}$ (e.g. Goppa).
- SK: code description $\Delta$ for $\mathcal{C}$.
- PK: parity-check matrix $H$ in systematic form for $\mathcal{C}$.

## Sign

- Compute $y = \mathbf{H}(msg)$.
- Set $e = Decode_\Delta(y)$, with $e \in \mathbb{F}_q^n$ of weight $w$.
- Signature is $\sigma = e$.

# The CFS Scheme

Select hash function $\mathbf{H} : \{0,1\}^* \to \mathbb{F}_q^{(n-k)}$.

## Key Generation

- Choose a code $\mathcal{C}$ (e.g. Goppa).
- SK: code description $\Delta$ for $\mathcal{C}$.
- PK: parity-check matrix $H$ in systematic form for $\mathcal{C}$.

## Sign

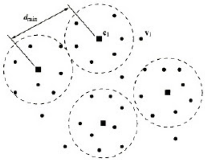
- Compute $y = \mathbf{H}(msg)$.
- Set $e = Decode_\Delta(y)$, with $e \in \mathbb{F}_q^n$ of weight $w$.
- Signature is $\sigma = e$.

## Verify

- Compute $y' = H\sigma^T$.
- Accept if $y' = \mathbf{H}(msg)$, otherwise reject.

# ABOUT CFS

Decoding a random word is not always possible!



# About CFS

Decoding a random word is not always possible!

Solution: try again.

# About CFS

Decoding a random word is not always possible!

Solution: try again.

Add counter $c$ to the input of $\mathbf{H}$ and try to decode $y = \mathbf{H}(msg, c)$.
Send $c$ along with signature.

# About CFS

Decoding a random word is not always possible!

Solution: try again.

Add counter $c$ to the input of $\mathbf{H}$ and try to decode $y = \mathbf{H}(msg, c)$.
Send $c$ along with signature.

With McEliece parameters $(1024, 524, 50)$ average of $2^{216}$ decoding attempts.

# About CFS

Decoding a random word is not always possible!

Solution: try again.

Add counter $c$ to the input of $\mathbf{H}$ and try to decode $y = \mathbf{H}(msg, c)$.
Send $c$ along with signature.

With McEliece parameters $(1024, 524, 50)$ average of $2^{216}$ decoding attempts.

Optimize parameters, e.g. $(2^{16}, 2^{16} - 144, 9)$ code: 9! decoding attempts.

# About CFS

Decoding a random word is not always possible!

Solution: try again.

Add counter $c$ to the input of $\mathbf{H}$ and try to decode $y = \mathbf{H}(msg, c)$.
Send $c$ along with signature.

With McEliece parameters $(1024, 524, 50)$ average of $2^{216}$ decoding attempts.

Optimize parameters, e.g. $(2^{16}, 2^{16} - 144, 9)$ code: 9! decoding attempts.

CFS parameters:

| $q$ | $m$ | $n$ | $w$ | PK (kB) | Sig (bits) | Security |
|---|---|---|---|---|---|---|
| 2 | 16 | 65536 | 9 | 1152 | 144 | 80 |

# About CFS

Decoding a random word is not always possible!

Solution: try again.

Add counter $c$ to the input of $\mathbf{H}$ and try to decode $y = \mathbf{H}(msg, c)$.
Send $c$ along with signature.

With McEliece parameters $(1024, 524, 50)$ average of $2^{216}$ decoding attempts.

Optimize parameters, e.g. $(2^{16}, 2^{16} - 144, 9)$ code: 9! decoding attempts.

CFS parameters:

| $q$ | $m$ | $n$ | $w$ | PK (kB) | Sig (bits) | Security |
|---|---|---|---|---|---|---|
| 2 | 16 | 65536 | 9 | 1152 | 144 | 80 |

Also, signing is very slow: in the order of seconds.

# Considerations

Additional security concerns: very high rate leads to distinguishers.

(Faugère Gauthier-Umana, Otmani, Perret, Tillich, 2013)

# Considerations

Additional security concerns: very high rate leads to distinguishers.
(Faugère Gauthier-Umana, Otmani, Perret, Tillich, 2013)

Generalized Birthday Attack by Bleichenbacher (2009) invalidates previous parameters: minimum $(m, w) = (15, 12)$ for 80 sec bits.

# Considerations

Additional security concerns: very high rate leads to distinguishers.

(Faugère Gauthier-Umana, Otmani, Perret, Tillich, 2013)

Generalized Birthday Attack by Bleichenbacher (2009) invalidates previous parameters: minimum $(m, w) = (15, 12)$ for 80 sec bits.

Completely impractical performance: implementations show GB of public key, and several seconds to sign.

(Landais, Sendrier, 2012; Bernstein, Chou, Schwabe, 2013 )

# Considerations

Additional security concerns: very high rate leads to distinguishers.
(Faugère Gauthier-Umana, Otmani, Perret, Tillich, 2013)

Generalized Birthday Attack by Bleichenbacher (2009) invalidates previous parameters: minimum $(m, w) = (15, 12)$ for 80 sec bits.

Completely impractical performance: implementations show GB of public key, and several seconds to sign.
(Landais, Sendrier, 2012; Bernstein, Chou, Schwabe, 2013 )

Recent approach using high-weight decoding presents an improvement (Debris-Alazard, Sendrier, Tillich, 2018).

# Considerations

Additional security concerns: very high rate leads to distinguishers.
(Faugère Gauthier-Umana, Otmani, Perret, Tillich, 2013)

Generalized Birthday Attack by Bleichenbacher (2009) invalidates previous parameters: minimum $(m, w) = (15, 12)$ for 80 sec bits.

Completely impractical performance: implementations show GB of public key, and several seconds to sign.
(Landais, Sendrier, 2012; Bernstein, Chou, Schwabe, 2013 )

Recent approach using high-weight decoding presents an improvement (Debris-Alazard, Sendrier, Tillich, 2018).

Still shows many similar features.

# Considerations

Additional security concerns: very high rate leads to distinguishers.
(Faugère Gauthier-Umana, Otmani, Perret, Tillich, 2013)

Generalized Birthday Attack by Bleichenbacher (2009) invalidates previous parameters: minimum $(m, w) = (15, 12)$ for 80 sec bits.

Completely impractical performance: implementations show GB of public key, and several seconds to sign.
(Landais, Sendrier, 2012; Bernstein, Chou, Schwabe, 2013 )

Recent approach using high-weight decoding presents an improvement (Debris-Alazard, Sendrier, Tillich, 2018).

Still shows many similar features.

Wave parameters:

| $q$ | $n$ | $k_u$ | $k_v$ | $w$ | PK (MB) | Sig (kB) | Security |
|---|---|---|---|---|---|---|---|
| 3 | 8492 | 3558 | 2047 | 7980 | 3.2 | 1.6 | 128 |

# Part III

## Zero-Knowledge Protocols

# Zero-Knowledge Identification Schemes

An interactive protocol to prove knowledge of a secret...

# Zero-Knowledge Identification Schemes

An interactive protocol to prove knowledge of a secret...

...without revealing anything about it.

# Zero-Knowledge Identification Schemes

An interactive protocol to prove knowledge of a secret...

...without revealing anything about it.

| Prover (SK) | | Verifier (PK) |
|---|---|---|
| Commit to random data | $\xrightarrow{\textit{com}}$ | |
| | $\xleftarrow{\textit{ch}}$ | Select random challenge |
| Produce response | $\xrightarrow{\textit{rsp}}$ | |
| | | Accept/Reject |

## Zero-Knowledge Identification Schemes

An interactive protocol to prove knowledge of a secret...

...without revealing anything about it.

| Prover (SK) | | Verifier (PK) |
|---|---|---|
| Commit to random data | $\xrightarrow{\quad com \quad}$ | |
| | $\xleftarrow{\quad ch \quad}$ | Select random challenge |
| Produce response | $\xrightarrow{\quad rsp \quad}$ | |
| | | Accept/Reject |

- Correctness: honest prover always gets accepted.

# Zero-Knowledge Identification Schemes

An interactive protocol to prove knowledge of a secret...

...without revealing anything about it.

| Prover (SK) | | Verifier (PK) |
|---|---|---|
| Commit to random data | $\xrightarrow{\textit{com}}$ | |
| | $\xleftarrow{\textit{ch}}$ | Select random challenge |
| Produce response | $\xrightarrow{\textit{rsp}}$ | |
| | | Accept/Reject |

- Correctness: honest prover always gets accepted.
- Soundness: dishonest prover (impersonator) has a bounded probability of succeeding.

# Zero-Knowledge Identification Schemes

An interactive protocol to prove knowledge of a secret...

...without revealing anything about it.

| **Prover (SK)** | | **Verifier (PK)** |
|---|---|---|
| Commit to random data | $\xrightarrow{\textit{com}}$ | |
| | $\xleftarrow{\textit{ch}}$ | Select random challenge |
| Produce response | $\xrightarrow{\textit{rsp}}$ | |
| | | Accept/Reject |

- Correctness: honest prover always gets accepted.
- Soundness: dishonest prover (impersonator) has a bounded probability of succeeding.
- Zero-Knowledge: no information about the secret is leaked.

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(com, msg)$.

# Building Signatures

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(\mathit{com}, \mathit{msg})$.
- Form signature as $\sigma = (\mathit{com}, \mathit{rsp})$.

# Building Signatures

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(\mathit{com}, \mathit{msg})$.
- Form signature as $\sigma = (\mathit{com}, \mathit{rsp})$.
- Verify as in identification protocol.

# Building Signatures

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(com, msg)$.
- Form signature as $\sigma = (com, rsp)$.
- Verify as in identification protocol.

---

This method for building signatures is very promising and leads to efficient schemes.

# Building Signatures

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(com, msg)$.
- Form signature as $\sigma = (com, rsp)$.
- Verify as in identification protocol.

---

This method for building signatures is very promising and leads to efficient schemes.

Strong security guarantees. No trapdoor is required!

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(com, msg)$.
- Form signature as $\sigma = (com, rsp)$.
- Verify as in identification protocol.

---

This method for building signatures is very promising and leads to efficient schemes.

Strong security guarantees. No trapdoor is required!

For code-based, can avoid decoding: rely directly on SDP.

# Building Signatures

ZKIDs can be turned into signature schemes using Fiat-Shamir transformation.

- Replace verifier's challenge with $\mathbf{H}(\textit{com}, \textit{msg})$.
- Form signature as $\sigma = (\textit{com}, \textit{rsp})$.
- Verify as in identification protocol.

---

This method for building signatures is very promising and leads to efficient schemes.

Strong security guarantees. No trapdoor is required!

For code-based, can avoid decoding: rely directly on SDP.

Use random codes and exploit hardness of finding low-weight words.
(Stern, 1993)

# Stern's ZKID Protocol

Select hash function **H**.

# Stern's ZKID Protocol

Select hash function **H**.

## Key Generation

- Choose random binary code $\mathcal{C}$, given by parity-check matrix $H$.
- SK: $e \in \mathbb{F}_2^n$ of weight $w$.
- PK: the syndrome $s = He^T$.

## STERN'S ZKID PROTOCOL

Select hash function **H**.

### KEY GENERATION

- Choose random binary code $\mathcal{C}$, given by parity-check matrix $H$.
- SK: $e \in \mathbb{F}_2^n$ of weight $w$.
- PK: the syndrome $s = He^T$.

### PROVER | VERIFIER

Choose $y \in \mathbb{F}_2^n$ and permutation $\pi$.
Set $c_1 = \mathbf{H}(\pi, Hy^T), c_2 = \mathbf{H}(\pi(y))$
$c_3 = \mathbf{H}(\pi(y + e))$

$$\xrightarrow{c_1, c_2, c_3}$$

$$\xleftarrow{b}$$ Select random $b \in \{0, 1, 2\}$.

If $b = 0$ set $rsp = (y, \pi)$ — Verify $c_1, c_2$.

If $b = 1$ set $rsp = (y + e, \pi)$ $\xrightarrow{rsp}$ — Verify $c_1, c_3$.

If $b = 2$ set $rsp = (\pi(y), \pi(e))$ — Verify $c_2, c_3$ and $wt(\pi(e)) = w$.

# About Stern's ZKID

High soundness error implies that adversary has non-trivial cheating probability.

# About Stern's ZKID

High soundness error implies that adversary has non-trivial cheating probability.

For Stern's scheme, soundness error is 2/3.

# About Stern's ZKID

High soundness error implies that adversary has non-trivial cheating probability.

For Stern's scheme, soundness error is 2/3.

For instance: choose random $y$ and $\pi$, then $x \in \mathbb{F}_2^n$ with $Hx^T = Hy^T + s$. Build $c_1$ and $c_2$ normally and $c_3 = \mathbf{H}(\pi(x))$ . Then $rsp = (y, \pi)$ and $rsp = (x, \pi)$ pass verification for $b = 0$ and $b = 1$ (strategy fails for $b = 2$). Similarly for other combinations.

# About Stern's ZKID

High soundness error implies that adversary has non-trivial cheating probability.

For Stern's scheme, soundness error is 2/3.

For instance: choose random $y$ and $\pi$, then $x \in \mathbb{F}_2^n$ with $Hx^T = Hy^T + s$. Build $c_1$ and $c_2$ normally and $c_3 = \mathbf{H}(\pi(x))$ . Then $rsp = (y, \pi)$ and $rsp = (x, \pi)$ pass verification for $b = 0$ and $b = 1$ (strategy fails for $b = 2$). Similarly for other combinations.

This means several repetitions are necessary to amplify error and reach target authentication level.

# About Stern's ZKID

High soundness error implies that adversary has non-trivial cheating probability.

For Stern's scheme, soundness error is 2/3.

For instance: choose random $y$ and $\pi$, then $x \in \mathbb{F}_2^n$ with $Hx^T = Hy^T + s$. Build $c_1$ and $c_2$ normally and $c_3 = \mathbf{H}(\pi(x))$ . Then $rsp = (y, \pi)$ and $rsp = (x, \pi)$ pass verification for $b = 0$ and $b = 1$ (strategy fails for $b = 2$). Similarly for other combinations.

This means several repetitions are necessary to amplify error and reach target authentication level.

Trasmitting the entire transcript produces a very long signature.

# About Stern's ZKID

High soundness error implies that adversary has non-trivial cheating probability.

For Stern's scheme, soundness error is 2/3.

For instance: choose random $y$ and $\pi$, then $x \in \mathbb{F}_2^n$ with $Hx^T = Hy^T + s$. Build $c_1$ and $c_2$ normally and $c_3 = \mathbf{H}(\pi(x))$ . Then $rsp = (y, \pi)$ and $rsp = (x, \pi)$ pass verification for $b = 0$ and $b = 1$ (strategy fails for $b = 2$). Similarly for other combinations.

This means several repetitions are necessary to amplify error and reach target authentication level.

Trasmitting the entire transcript produces a very long signature.

Stern's ZKID parameters:

| $q$ | $n$ | $w$ | $\tau$ | PK (bits) | Sig (kB) | Security | Auth. |
|---|---|---|---|---|---|---|---|
| 2 | 512 | 56 | 35 | 256 | 5 | 60 | 20 |
| 2 | 620 | 68 | 137 | 310 | 93.3 | 80 | 80 |
| 2 | 1024 | 112 | 219 | 512 | 245 | 128 | 128 |

# Considerations

Several variants proposed over the years:

# Considerations

Several variants proposed over the years:

- Stern, 1993.
- Véron, 1996.
- Gaborit, Girault, 2007.
- Cayrel, Véron, El Yousfi, 2010.
- Aguilar, Gaborit, Schrek, 2011.
- ...

# Considerations

Several variants proposed over the years:

- Stern, 1993.
- Véron, 1996.
- Gaborit, Girault, 2007.
- Cayrel, Véron, El Yousfi, 2010.
- Aguilar, Gaborit, Schrek, 2011.
- ...

Goal: decreasing soundness error.

# CONSIDERATIONS

Several variants proposed over the years:

- Stern, 1993.
- Véron, 1996.
- Gaborit, Girault, 2007.
- Cayrel, Véron, El Yousfi, 2010.
- Aguilar, Gaborit, Schrek, 2011.
- ...

Goal: decreasing soundness error.

For example, CVE scheme achieves $\frac{q}{2(q-1)} \approx 1/2$.

# CONSIDERATIONS

Several variants proposed over the years:

- Stern, 1993.
- Véron, 1996.
- Gaborit, Girault, 2007.
- Cayrel, Véron, El Yousfi, 2010.
- Aguilar, Gaborit, Schrek, 2011.
- ...

Goal: decreasing soundness error.

For example, CVE scheme achieves $\dfrac{q}{2(q-1)} \approx 1/2$.

Efficient for large finite fields.

Select hash function **H**.

Select hash function **H**.

### Key Generation

- Choose random $q$-ary code $\mathcal{C}$, given by parity-check matrix $H$.
- SK: $e \in \mathbb{F}_q^n$ of weight $w$.
- PK: the syndrome $s = He^T$.

# CVE's Protocol

Select hash function **H**.

## Key Generation

- Choose random $q$-ary code $\mathcal{C}$, given by parity-check matrix $H$.
- SK: $e \in \mathbb{F}_q^n$ of weight $w$.
- PK: the syndrome $s = He^T$.

## Prover / Verifier

| Prover | | Verifier |
|---|---|---|
| Choose $y \in \mathbb{F}_q^n$ and monomial $\mu$. Set $c_1 = \mathbf{H}(\mu, Hy^T)$, $c_2 = \mathbf{H}(\mu(y), \mu(e))$ | $\xrightarrow{c_1, c_2}$ | |
| | $\xleftarrow{c}$ | Select random $c \in \mathbb{F}_q^*$. |
| $z = \mu(y + ce)$ | $\xrightarrow{z}$ | |
| | $\xleftarrow{b}$ | Select random $b \in \{0, 1\}$. |
| If $b = 0$ set $rsp = \mu$ | | Verify $c_1 = \mathbf{H}(\mu, H\mu^{-1}(z))^T - cs)$. |
| If $b = 1$ set $rsp = \mu(e)$ | $\xrightarrow{rsp}$ | Verify $c_2 = \mathbf{H}(z - c\mu(e), \mu(e))$ and $wt(\mu(e)) = w$. |

# Part IV

## New Frameworks

# Decreasing the Soundness Error

Signature sizes still too large ($>$ 30 kB).

# DECREASING THE SOUNDNESS ERROR

Signature sizes still too large ($>$ 30 kB).

The idea is still valid - need to use a different protocol.

# Decreasing the Soundness Error

Signature sizes still too large ($> 30$ kB).

The idea is still valid - need to use a different protocol.

"MPC-in-the-head" approach used e.g. in Picnic.

(Ishai, Kushilevitz, Ostrovsky, Sahai, 2007; Katz, Kolesnikov, Wang, 2018)

# Decreasing the Soundness Error

Signature sizes still too large ($> 30$ kB).

The idea is still valid - need to use a different protocol.

“MPC-in-the-head” approach used e.g. in Picnic.
(Ishai, Kushilevitz, Ostrovsky, Sahai, 2007; Katz, Kolesnikov, Wang, 2018)

Some of the verifier’s checks are independent from the secret.

# Decreasing the Soundness Error

Signature sizes still too large ($> 30$ kB).

The idea is still valid - need to use a different protocol.

"MPC-in-the-head" approach used e.g. in Picnic.
(Ishai, Kushilevitz, Ostrovsky, Sahai, 2007; Katz, Kolesnikov, Wang, 2018)

Some of the verifier's checks are independent from the secret.

These can be offloaded to a trusted setup ("helper").

# Decreasing the Soundness Error

Signature sizes still too large ($> 30$ kB).

The idea is still valid - need to use a different protocol.

“MPC-in-the-head” approach used e.g. in Picnic.
(Ishai, Kushilevitz, Ostrovsky, Sahai, 2007; Katz, Kolesnikov, Wang, 2018)

Some of the verifier’s checks are independent from the secret.

These can be offloaded to a trusted setup (“helper”).

Preprocessing phase prepares auxiliary collection of samples (shares).

# Decreasing the Soundness Error

Signature sizes still too large ($>$ 30 kB).

The idea is still valid - need to use a different protocol.

"MPC-in-the-head" approach used e.g. in Picnic.
(Ishai, Kushilevitz, Ostrovsky, Sahai, 2007; Katz, Kolesnikov, Wang, 2018)

Some of the verifier's checks are independent from the secret.

These can be offloaded to a trusted setup ("helper").

Preprocessing phase prepares auxiliary collection of samples (shares).

"Opening" of a subset does not compromise security...

# Decreasing the Soundness Error

Signature sizes still too large ($>$ 30 kB).

The idea is still valid - need to use a different protocol.

"MPC-in-the-head" approach used e.g. in Picnic.
(Ishai, Kushilevitz, Ostrovsky, Sahai, 2007; Katz, Kolesnikov, Wang, 2018)

Some of the verifier's checks are independent from the secret.

These can be offloaded to a trusted setup ("helper").

Preprocessing phase prepares auxiliary collection of samples (shares).

"Opening" of a subset does not compromise security...

...but allows for much larger challenge space.

# GPS Protocol

KeyGen: as in CVE, using a commitment scheme **Com**.

KeyGen: as in CVE, using a commitment scheme **Com**.

## Helper

- Generate random $y, \tilde{e} \in \mathbb{F}_q^n$, with $\tilde{e}$ of weight $w$, from seed.
- Compute $aux = \{\mathbf{Com}(y + c\tilde{e})\}_{c \in \mathbb{F}_q}$.
- Send seed to prover and aux to verifier.

# GPS Protocol

KeyGen: as in CVE, using a commitment scheme **Com**.

### Helper

- Generate random $y, \tilde{e} \in \mathbb{F}_q^n$, with $\tilde{e}$ of weight $w$, from seed.
- Compute $aux = \{\mathbf{Com}(y + c\tilde{e})\}_{c \in \mathbb{F}_q}$.
- Send seed to prover and aux to verifier.

### Prover / Verifier

| Prover | | Verifier |
|---|---|---|
| Regenerate $y, \tilde{e}$ from seed. | | |
| Determine $\mu$ s.t. $e = \mu(\tilde{e})$ | | |
| $\alpha = \mathbf{Com}(\mu, H(\mu(y))^T)$ | $\xrightarrow{\alpha}$ | |
| | $\xleftarrow{c}$ | Select random $c \in \mathbb{F}_q$. |
| $z = y + c\tilde{e}$ | $\xrightarrow{z}$ | |
| | | Verify $\alpha = \mathbf{Com}(\mu, H(\mu(z))^T - cs)$. |
| | | Verify $\mathbf{Com}(z)$ with corresponding value from $aux$. |

# GPS Protocol

KeyGen: as in CVE, using a commitment scheme **Com**.

### Helper

- Generate random $y, \tilde{e} \in \mathbb{F}_q^n$, with $\tilde{e}$ of weight $w$, from seed.
- Compute $aux = \{\mathbf{Com}(y + c\tilde{e})\}_{c \in \mathbb{F}_q}$.
- Send seed to prover and aux to verifier.

### Prover — Verifier

Regenerate $y, \tilde{e}$ from seed.

Determine $\mu$ s.t. $e = \mu(\tilde{e})$

$\alpha = \mathbf{Com}(\mu, H(\mu(y))^T)$ $\xrightarrow{\alpha}$

$\xleftarrow{c}$ Select random $c \in \mathbb{F}_q$.

$z = y + c\tilde{e}$ $\xrightarrow{z}$

Verify $\alpha = \mathbf{Com}(\mu, H(\mu(z))^T - cs)$.

Verify $\mathbf{Com}(z)$ with corresponding value from $aux$.

Here the soundness error is $1/q$.

# Producing a Signature Scheme

Use "cut-and-choose" technique to remove preprocessing.

# Producing a Signature Scheme

Use “cut-and-choose” technique to remove preprocessing.

Executing 1 out of $M$ setups produces soundness error $max\left(\frac{1}{M}, \frac{1}{q}\right)$.

# Producing a Signature Scheme

Use “cut-and-choose” technique to remove preprocessing.

Executing 1 out of $M$ setups produces soundness error $max\left(\frac{1}{M}, \frac{1}{q}\right)$.

Iterate as needed, then apply Fiat-Shamir.

# Producing a Signature Scheme

Use "cut-and-choose" technique to remove preprocessing.

Executing 1 out of $M$ setups produces soundness error $max\left(\frac{1}{M}, \frac{1}{q}\right)$.

Iterate as needed, then apply Fiat-Shamir.

Several optimizations are possible (e.g. Merkle trees, seed trees).

# Producing a Signature Scheme

Use “cut-and-choose” technique to remove preprocessing.

Executing 1 out of $M$ setups produces soundness error $max\left(\frac{1}{M}, \frac{1}{q}\right)$.

Iterate as needed, then apply Fiat-Shamir.

Several optimizations are possible (e.g. Merkle trees, seed trees).

Can potentially yield smaller signatures, at the cost of increased computation (signing/verification time).

# Producing a Signature Scheme

Use “cut-and-choose” technique to remove preprocessing.

Executing 1 out of $M$ setups produces soundness error $max\left(\frac{1}{M}, \frac{1}{q}\right)$.

Iterate as needed, then apply Fiat-Shamir.

Several optimizations are possible (e.g. Merkle trees, seed trees).

Can potentially yield smaller signatures, at the cost of increased computation (signing/verification time).

GPS scheme parameters ($\lambda = 128$, sizes in kB):

| $M$ | $\tau$ | $q$ | $n$ | $k$ | $w$ | PK | Sig |
|---|---|---|---|---|---|---|---|
| 512 | 23 | 128 | 220 | 101 | 90 | 0.10 | 27.06 |
| 1024 | 19 | 256 | 207 | 93 | 90 | 0.11 | 23.98 |
| 2048 | 16 | 512 | 196 | 92 | 84 | 0.11 | 21.22 |
| 4096 | 14 | 1024 | 187 | 90 | 80 | 0.12 | 19.76 |

# Shared Permutations

The GPS machinery is very expensive.

# Shared Permutations

The GPS machinery is very expensive.

Idea: use MPC-in-the-head on permutations.

(Feneuil, Joux, Rivain, 2021)

# Shared Permutations

The GPS machinery is very expensive.

Idea: use MPC-in-the-head on permutations.
(Feneuil, Joux, Rivain, 2021)

As in Stern, reveal $v := \pi(e)$ and $\tilde{e} := e + y$, for codeword $y$.
Verifier then checks $s = H\tilde{e}^T$ and $wt(v)$.

# Shared Permutations

The GPS machinery is very expensive.

Idea: use MPC-in-the-head on permutations.
(Feneuil, Joux, Rivain, 2021)

As in Stern, reveal $v := \pi(e)$ and $\tilde{e} := e + y$, for codeword $y$.
Verifier then checks $s = H\tilde{e}^T$ and $wt(v)$.

This is equivalent to showing existence of $(\pi, y)$ such that $Hy^T = 0$ and $\pi(\tilde{e}) = v + \pi(y)$.

# SHARED PERMUTATIONS

The GPS machinery is very expensive.

Idea: use MPC-in-the-head on permutations.
(Feneuil, Joux, Rivain, 2021)

As in Stern, reveal $v := \pi(e)$ and $\tilde{e} := e + y$, for codeword $y$.
Verifier then checks $s = H\tilde{e}^T$ and $wt(v)$.

This is equivalent to showing existence of $(\pi, y)$ such that $Hy^T = 0$ and $\pi(\tilde{e}) = v + \pi(y)$.

Can prove the former using cut-and-choose, and the latter via an affine transformation $A(\cdot) = \pi(\cdot) + r$, for random $r$, so that

$$A(\tilde{e}) = \pi(e) + \pi(y) + r = v + q$$

where $q$ is a “trusted” vector.

# Shared Permutations

The GPS machinery is very expensive.

Idea: use MPC-in-the-head on permutations.
(Feneuil, Joux, Rivain, 2021)

As in Stern, reveal $v := \pi(e)$ and $\tilde{e} := e + y$, for codeword $y$.
Verifier then checks $s = H\tilde{e}^T$ and $wt(v)$.

This is equivalent to showing existence of $(\pi, y)$ such that $Hy^T = 0$ and $\pi(\tilde{e}) = v + \pi(y)$.

Can prove the former using cut-and-choose, and the latter via an affine transformation $A(\cdot) = \pi(\cdot) + r$, for random $r$, so that

$$A(\tilde{e}) = \pi(e) + \pi(y) + r = v + q$$

where $q$ is a “trusted” vector.

$A$ is decomposed into $A_N \circ \cdots \circ A_1$ to apply MPC.

# Shared Permutations

The GPS machinery is very expensive.

Idea: use MPC-in-the-head on permutations.
(Feneuil, Joux, Rivain, 2021)

As in Stern, reveal $v := \pi(e)$ and $\tilde{e} := e + y$, for codeword $y$.
Verifier then checks $s = H\tilde{e}^T$ and $wt(v)$.

This is equivalent to showing existence of $(\pi, y)$ such that $Hy^T = 0$ and $\pi(\tilde{e}) = v + \pi(y)$.

Can prove the former using cut-and-choose, and the latter via an affine transformation $A(\cdot) = \pi(\cdot) + r$, for random $r$, so that

$$A(\tilde{e}) = \pi(e) + \pi(y) + r = v + q$$

where $q$ is a "trusted" vector.

$A$ is decomposed into $A_N \circ \cdots \circ A_1$ to apply MPC.
FJR scheme parameters ($\lambda = 128$, sizes in kB):

| $M$ | $\tau$ | $q$ | $n$ | $k$ | $w$ | PK | Sig |
|---|---|---|---|---|---|---|---|
| 389 | 28 | 2 | 1280 | 640 | 132 | 0.96 | 16.34 |

# Proving Hamming Weight via Polynomials

Observation: if $H = (H'|I_{n-k})$ write $e = (e_A, e_B)$, so $s = H(e_A, e_B)^T$.
Then $e_A$ uniquely determines $e$ given $s$ and $H$.

# Proving Hamming Weight via Polynomials

Observation: if $H = (H'|I_{n-k})$ write $e = (e_A, e_B)$, so $s = H(e_A, e_B)^T$. Then $e_A$ uniquely determines $e$ given $s$ and $H$.

Let $\mathbb{F}_q \subset \mathbb{F}_{\text{poly}}$ such that $n \leq |\mathbb{F}_{\text{poly}}|$ and let $\{\gamma_1, \ldots, \gamma_n\}$ be distinct elements of $\mathbb{F}_{\text{poly}}$.

# Proving Hamming Weight via Polynomials

Observation: if $H = (H'|I_{n-k})$ write $e = (e_A, e_B)$, so $s = H(e_A, e_B)^T$. Then $e_A$ uniquely determines $e$ given $s$ and $H$.

Let $\mathbb{F}_q \subset \mathbb{F}_{\text{poly}}$ such that $n \leq |\mathbb{F}_{\text{poly}}|$ and let $\{\gamma_1, \ldots, \gamma_n\}$ be distinct elements of $\mathbb{F}_{\text{poly}}$.

Build $S(X) \in \mathbb{F}_{\text{poly}}[X]$ via polynomial interpolation of the points $(\gamma_i, e_i)$.

# PROVING HAMMING WEIGHT VIA POLYNOMIALS

Observation: if $H = (H'|I_{n-k})$ write $e = (e_A, e_B)$, so $s = H(e_A, e_B)^T$. Then $e_A$ uniquely determines $e$ given $s$ and $H$.

Let $\mathbb{F}_q \subset \mathbb{F}_{\text{poly}}$ such that $n \leq |\mathbb{F}_{\text{poly}}|$ and let $\{\gamma_1, \ldots, \gamma_n\}$ be distinct elements of $\mathbb{F}_{\text{poly}}$.

Build $S(X) \in \mathbb{F}_{\text{poly}}[X]$ via polynomial interpolation of the points $(\gamma_i, e_i)$.

Build $Q(X) = \prod_{i \in E}(X - \gamma_i)$ where $E =$\{nonzero pos. of $e$\}.

# Proving Hamming Weight via Polynomials

Observation: if $H = (H'|I_{n-k})$ write $e = (e_A, e_B)$, so $s = H(e_A, e_B)^T$. Then $e_A$ uniquely determines $e$ given $s$ and $H$.

Let $\mathbb{F}_q \subset \mathbb{F}_{\text{poly}}$ such that $n \leq |\mathbb{F}_{\text{poly}}|$ and let $\{\gamma_1, \ldots, \gamma_n\}$ be distinct elements of $\mathbb{F}_{\text{poly}}$.

Build $S(X) \in \mathbb{F}_{\text{poly}}[X]$ via polynomial interpolation of the points $(\gamma_i, e_i)$.

Build $Q(X) = \prod_{i \in E}(X - \gamma_i)$ where $E =$\{nonzero pos. of $e$\}.

Then $deg(Q) = w$ and $wt(e) \leq w$ is equivalent to

$$Q \cdot S - P \cdot F = 0$$

where $F = \prod_{i=1}^{n}(X - \gamma_i)$ and $deg(P) \leq w - 1$.

Observation: if $H = (H'|I_{n-k})$ write $e = (e_A, e_B)$, so $s = H(e_A, e_B)^T$. Then $e_A$ uniquely determines $e$ given $s$ and $H$.

Let $\mathbb{F}_q \subset \mathbb{F}_{\mathrm{poly}}$ such that $n \leq |\mathbb{F}_{\mathrm{poly}}|$ and let $\{\gamma_1, \ldots, \gamma_n\}$ be distinct elements of $\mathbb{F}_{\mathrm{poly}}$.

Build $S(X) \in \mathbb{F}_{\mathrm{poly}}[X]$ via polynomial interpolation of the points $(\gamma_i, e_i)$.

Build $Q(X) = \prod_{i \in E}(X - \gamma_i)$ where $E =$\{nonzero pos. of $e$\}.

Then $deg(Q) = w$ and $wt(e) \leq w$ is equivalent to

$$Q \cdot S - P \cdot F = 0$$

where $F = \prod_{i=1}^{n}(X - \gamma_i)$ and $deg(P) \leq w - 1$.

This transforms SDP into a polynomial problem and completely avoids the need for an isometry.

(Feneuil, Joux, Rivain, 2022)

# Proof of Knowledge (in a nutshell)

Create some shares $e_A = \sum_{j=1}^{M} e_A^{(j)}$ and hence $e = \sum_{j=1}^{M} e^{(j)}$.

# Proof of Knowledge (in a nutshell)

Create some shares $e_A = \sum_{j=1}^{M} e_A^{(j)}$ and hence $e = \sum_{j=1}^{M} e^{(j)}$.

Find $S^{(j)}(X)$ using points $(\gamma_i, e_i^{(j)})$, where $e^{(j)} = (e_1^{(j)}, \ldots, e_n^{(j)})$.

Create some shares $e_A = \sum_{j=1}^{M} e_A^{(j)}$ and hence $e = \sum_{j=1}^{M} e^{(j)}$.

Find $S^{(j)}(X)$ using points $(\gamma_i, e_i^{(j)})$, where $e^{(j)} = (e_1^{(j)}, \ldots, e_n^{(j)})$.

By the linearity of the Lagrange interpolation, $S(X) = \sum_{j=1}^{M} S^{(j)}(X)$.

# Proof of Knowledge (in a nutshell)

Create some shares $e_A = \sum_{j=1}^{M} e_A^{(j)}$ and hence $e = \sum_{j=1}^{M} e^{(j)}$.

Find $S^{(j)}(X)$ using points $(\gamma_i, e_i^{(j)})$, where $e^{(j)} = (e_1^{(j)}, \ldots, e_n^{(j)})$.

By the linearity of the Lagrange interpolation, $S(X) = \sum_{j=1}^{M} S^{(j)}(X)$.

Write $Q(X) = \sum_{j=1}^{M} Q^{(j)}(X)$ and then $(P \cdot F)(X) = \sum_{j=1}^{M} (P \cdot F)^{(j)}(X)$.

# Proof of Knowledge (in a nutshell)

Create some shares $e_A = \sum_{j=1}^{M} e_A^{(j)}$ and hence $e = \sum_{j=1}^{M} e^{(j)}$.

Find $S^{(j)}(X)$ using points $(\gamma_i, e_i^{(j)})$, where $e^{(j)} = (e_1^{(j)}, \dots, e_n^{(j)})$.

By the linearity of the Lagrange interpolation, $S(X) = \sum_{j=1}^{M} S^{(j)}(X)$.

Write $Q(X) = \sum_{j=1}^{M} Q^{(j)}(X)$ and then $(P \cdot F)(X) = \sum_{j=1}^{M} (P \cdot F)^{(j)}(X)$.

To verify that $Q(X)S(X) = (P \cdot F)(X)$, check $Q(r_l)S(r_l) = (P \cdot F)(r_l)$ for $1 \leq l \leq t$ and $r_l$ elements of an extension field $\mathbb{F}_{\text{points}}$ of $\mathbb{F}_{\text{poly}}$ (Schwartz-Zippel lemma).

# PROOF OF KNOWLEDGE (IN A NUTSHELL)

Create some shares $e_A = \sum_{j=1}^{M} e_A^{(j)}$ and hence $e = \sum_{j=1}^{M} e^{(j)}$.

Find $S^{(j)}(X)$ using points $(\gamma_i, e_i^{(j)})$, where $e^{(j)} = (e_1^{(j)}, \ldots, e_n^{(j)})$.

By the linearity of the Lagrange interpolation, $S(X) = \sum_{j=1}^{M} S^{(j)}(X)$.

Write $Q(X) = \sum_{j=1}^{M} Q^{(j)}(X)$ and then $(P \cdot F)(X) = \sum_{j=1}^{M} (P \cdot F)^{(j)}(X)$.

To verify that $Q(X)S(X) = (P \cdot F)(X)$, check $Q(r_l)S(r_l) = (P \cdot F)(r_l)$ for $1 \leq l \leq t$ and $r_l$ elements of an extension field $\mathbb{F}_{\text{points}}$ of $\mathbb{F}_{\text{poly}}$ (Schwartz-Zippel lemma).

This is done directly on shares $Q^{(j)}(r_l)$, $S^{(j)}(r_l)$ and $(P \cdot F)^{(j)}(r_l)$, via standard MPC techniques to verify multiplication triple.

Signature scheme obtained via usual means (cut-and-choose, repetition, Fiat-Shamir).

# Considerations and Future Work

Signature scheme obtained via usual means (cut-and-choose, repetition, Fiat-Shamir).

Performance is extremely competitive!

# Considerations and Future Work

Signature scheme obtained via usual means (cut-and-choose, repetition, Fiat-Shamir).

Performance is extremely competitive!

Scheme parameters ($\lambda = 128$, sizes in kB):

| $M$ | $\tau$ | $q$ | $n$ | $k$ | $w$ | $\mathbb{F}_{\text{poly}}$ | $\mathbb{F}_{\text{points}}$ | PK | Sig |
|---|---|---|---|---|---|---|---|---|---|
| 256 | 17 | 2 | 1280 | 640 | 132 | $2^{11}$ | $2^{22}$ | 0.96 | 11.2 |
| 256 | 17 | $2^8$ | 256 | 128 | 80 | $2^8$ | $2^{24}$ | 0.15 | 8.5 |

# Considerations and Future Work

Signature scheme obtained via usual means (cut-and-choose, repetition, Fiat-Shamir).

Performance is extremely competitive!

Scheme parameters ($\lambda = 128$, sizes in kB):

| $M$ | $\tau$ | $q$ | $n$ | $k$ | $w$ | $\mathbb{F}_{\text{poly}}$ | $\mathbb{F}_{\text{points}}$ | PK | Sig |
|---|---|---|---|---|---|---|---|---|---|
| 256 | 17 | 2 | 1280 | 640 | 132 | $2^{11}$ | $2^{22}$ | 0.96 | 11.2 |
| 256 | 17 | $2^8$ | 256 | 128 | 80 | $2^8$ | $2^{24}$ | 0.15 | 8.5 |

Possible improvement using linear complexity to avoid interpolation.

(P., Randrianarisoa, 2022)

# Considerations and Future Work

Signature scheme obtained via usual means (cut-and-choose, repetition, Fiat-Shamir).

Performance is extremely competitive!

Scheme parameters ($\lambda = 128$, sizes in kB):

| $M$ | $\tau$ | $q$ | $n$ | $k$ | $w$ | $\mathbb{F}_{\text{poly}}$ | $\mathbb{F}_{\text{points}}$ | PK | Sig |
|---|---|---|---|---|---|---|---|---|---|
| 256 | 17 | 2 | 1280 | 640 | 132 | $2^{11}$ | $2^{22}$ | 0.96 | 11.2 |
| 256 | 17 | $2^8$ | 256 | 128 | 80 | $2^8$ | $2^{24}$ | 0.15 | 8.5 |

Possible improvement using linear complexity to avoid interpolation.
(P., Randrianarisoa, 2022)

$q$-ary parameters can be refined, leading to improved performance (e.g. Sig $\approx$ 7 kB).

# Considerations and Future Work

Signature scheme obtained via usual means (cut-and-choose, repetition, Fiat-Shamir).

Performance is extremely competitive!

Scheme parameters ($\lambda = 128$, sizes in kB):

| $M$ | $\tau$ | $q$ | $n$ | $k$ | $w$ | $\mathbb{F}_{\text{poly}}$ | $\mathbb{F}_{\text{points}}$ | PK | Sig |
|---|---|---|---|---|---|---|---|---|---|
| 256 | 17 | 2 | 1280 | 640 | 132 | $2^{11}$ | $2^{22}$ | 0.96 | 11.2 |
| 256 | 17 | $2^8$ | 256 | 128 | 80 | $2^8$ | $2^{24}$ | 0.15 | 8.5 |

Possible improvement using linear complexity to avoid interpolation.
(P., Randrianarisoa, 2022)

*q*-ary parameters can be refined, leading to improved performance (e.g. Sig $\approx$ 7 kB).

Optimized implementation underway, NIST submission on the horizon.

# Part V

## Conclusions

Finally, viable solutions for code-based signatures begin to appear.

# Intuition

Finally, viable solutions for code-based signatures begin to appear.

MPC-in-the-head approach allows to achieve high soundness, bypassing traditional Achilles' heel of ZK-based schemes.

# Intuition

Finally, viable solutions for code-based signatures begin to appear.

MPC-in-the-head approach allows to achieve high soundness, bypassing traditional Achilles' heel of ZK-based schemes.

Another line of research uses isometries to build efficient schemes in a completely different way: for code-based, LESS (code equivalence).
(Biasse, Micheli, P. Santini, 2020; Barenghi, Biasse, P., Santini, 2021)

# Intuition

Finally, viable solutions for code-based signatures begin to appear.

MPC-in-the-head approach allows to achieve high soundness, bypassing traditional Achilles' heel of ZK-based schemes.

Another line of research uses isometries to build efficient schemes in a completely different way: for code-based, LESS (code equivalence).
(Biasse, Micheli, P. Santini, 2020; Barenghi, Biasse, P., Santini, 2021)

A lot of work to do for designing, optimizing and cryptanalyzing such schemes.

# Intuition

Finally, viable solutions for code-based signatures begin to appear.

MPC-in-the-head approach allows to achieve high soundness, bypassing traditional Achilles' heel of ZK-based schemes.

Another line of research uses isometries to build efficient schemes in a completely different way: for code-based, LESS (code equivalence).
(Biasse, Micheli, P. Santini, 2020; Barenghi, Biasse, P., Santini, 2021)

A lot of work to do for designing, optimizing and cryptanalyzing such schemes.

Investigate applicability to several advanced frameworks (e.g. ring sigs, identity-based sigs, threshold sigs, multi-sigs...)

Grazie per l'attenzione!