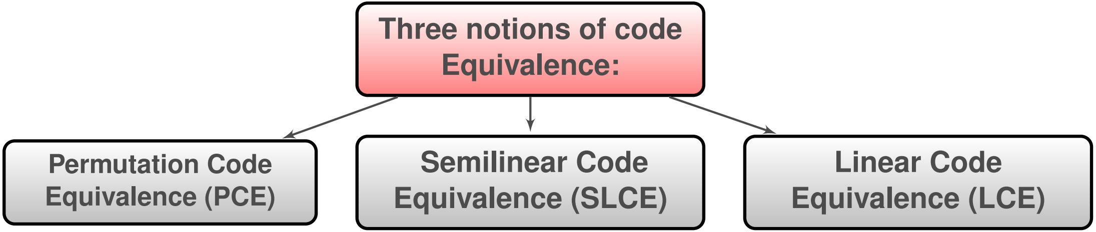
Code-Based Cryptography

1. Error-Correcting Codes and Cryptography
2. McEliece Cryptosystem
3. Message Attacks (ISD)
4. **Key Attacks**
5. Other Cryptographic Constructions Relying on Coding Theory

4. Key Attacks

The Code Equivalence Problem of Linear Codes



The Code Equivalence Problem of Linear Codes

Three notions of code Equivalence:

↓

Semilinear Code Equivalence (SLCE)

Semilinear Code Equivalence (SLE)

$$\mathcal{C}_1 \overset{\text{SLE}}{\sim} \mathcal{C}_2 \iff \exists \phi : \mathcal{C}_2 = \phi(\mathcal{C}_1)$$

$$\text{with } \phi = (\underbrace{\sigma \in S_n}_{\text{Permutation}}, \underbrace{\lambda = (\lambda_1, \ldots, \lambda_n) \in (\mathbb{F}_q^*)^n}_{\textit{Scalar}}, \underbrace{\alpha \in \mathrm{Aut}(\mathbb{F}_q)}_{\textit{Automorphism}})$$

The Code Equivalence Problem of Linear Codes

Three notions of code Equivalence:

Permutation Code Equivalence (PCE)

Permutation Code Equivalence (PE)

$$\mathcal{C}_1 \sim \mathcal{C}_2 \iff \exists \underbrace{\sigma \in S_n}_{\text{Permutation}} : \mathcal{C}_2 = \sigma(\mathcal{C}_1) = \{\sigma(\mathbf{x}) \mid \mathbf{x} \in \mathcal{C}\}$$

The Code Equivalence Problem of Linear Codes

Three notions of code Equivalence:

- Permutation Code Equivalence (PCE)
- Semilinear Code Equivalence (SLCE)
- Linear Code Equivalence (LCE)

Permutation Code Equivalence (PE)

$$\mathcal{C}_1 \sim \mathcal{C}_2 \iff \exists \underbrace{\sigma \in S_n}_{\text{Permutation}} : \mathcal{C}_2 = \sigma(\mathcal{C}_1) = \{\sigma(\mathbf{x}) \mid \mathbf{x} \in \mathcal{C}\}$$

The Code Equivalence Problem of Linear Codes

The Code Equivalence Problem

INPUT: Two $[n, k]_q$ linear codes: $\mathcal{C}_1$ and $\mathcal{C}_2$

The Code Equivalence Problem of Linear Codes

The Code Equivalence Problem

INPUT: Two $[n, k]_q$ linear codes: $\mathcal{C}_1$ and $\mathcal{C}_2$

OUTPUT:

The Code Equivalence Problem of Linear Codes

The Code Equivalence Problem

INPUT: Two $[n, k]_q$ linear codes: $\mathcal{C}_1$ and $\mathcal{C}_2$

OUTPUT:

(Decision): Are $\mathcal{C}_1 \sim \mathcal{C}_2$?

The Code Equivalence Problem of Linear Codes

The Code Equivalence Problem

INPUT: Two $[n, k]_q$ linear codes: $\mathcal{C}_1$ and $\mathcal{C}_2$

OUTPUT:

(Decision): Are $\mathcal{C}_1 \sim \mathcal{C}_2$?

(Computational): If $\mathcal{C}_1 \sim \mathcal{C}_2$. Find $\sigma \in S_n$ such that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$

The Code Equivalence Problem of Linear Codes

➜ **Complexity:** The **PE** problem is not NP-Complete but it is at least as hard as **Graph Isomorphism Problem**

E. Petrank and R.M. Roth,
Is code equivalence easy to decide?,
1997.

The Code Equivalence Problem of Linear Codes

➜ **Complexity:** The **PE** problem is not NP-Complete but it is at least as hard as **Graph Isomorphism Problem**

E. Petrank and R.M. Roth,
Is code equivalence easy to decide?,
1997.

➜ **Known Algorithms:**

- The **Support Splitting Algorithm** for PE for $\mathbb{F}_2$, $\mathbb{F}_3$ and $\mathbb{F}_4$

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

N. Sendrier and D. E. Simos
The hardness of code equivalence over $\mathbb{F}_q$ and its application to code-based cryptography.
Post-Quantum Cryptography, volume 7932 of LNCS, 203-216, 2013.

The Code Equivalence Problem of Linear Codes

- **Complexity:** The **PE** problem is not NP-Complete but it is at least as hard as **Graph Isomorphism Problem**

  E. Petrank and R.M. Roth,
  Is code equivalence easy to decide?,
  1997.

- **Known Algorithms:**

  - The **Support Splitting Algorithm** for PE for $\mathbb{F}_2$, $\mathbb{F}_3$ and $\mathbb{F}_4$

    N. Sendrier,
    Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
    IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

    N. Sendrier and D. E. Simos
    The hardness of code equivalence over $\mathbb{F}_q$ and its application to code-based cryptography.
    Post-Quantum Cryptography, volume 7932 of LNCS, 203-216, 2013.

  - **Computation of canonical forms** for LC over $\mathbb{F}_q$, with q small.

    T. Feulner,
    The automorphism groups of linear codes and canonical representatives of their semilinear isometry classes,
    AMC, vol. 3 (4), p. 363-383, 2009

Invariants

Invariants

$\mathcal{V}$ is an **invariant** if $\mathcal{C}_1 \sim \mathcal{C}_2 \Rightarrow \mathcal{V}(\mathcal{C}_1) = \mathcal{V}(\mathcal{C}_2)$

Invariants

Invariants

$\mathcal{V}$ is an **invariant** if $\mathcal{C}_1 \sim \mathcal{C}_2 \Rightarrow \mathcal{V}(\mathcal{C}_1) = \mathcal{V}(\mathcal{C}_2)$

The Weight Enumerator is an invariant: $\mathcal{C}_1 \sim \mathcal{C}_2 \Rightarrow \mathcal{W}_{\mathcal{C}_1}(X) = \mathcal{W}_{\mathcal{C}_2}(X)$

Recall that $\mathcal{W}_{\mathcal{C}}(X) = \sum_{i=0}^{n} A_i X^i$ with $A_i = |\{\mathbf{c} \in \mathcal{C} \mid \mathrm{w}_H(\mathbf{c}) = i\}|$

Invariants

Invariants

$\mathcal{V}$ is an **invariant** if $\mathcal{C}_1 \sim \mathcal{C}_2 \Rightarrow \mathcal{V}(\mathcal{C}_1) = \mathcal{V}(\mathcal{C}_2)$

The Weight Enumerator is an invariant: $\mathcal{C}_1 \sim \mathcal{C}_2 \Rightarrow \mathcal{W}_{\mathcal{C}_1}(X) = \mathcal{W}_{\mathcal{C}_2}(X)$

Recall that $\mathcal{W}_{\mathcal{C}}(X) = \sum_{i=0}^{n} A_i X^i$ with $A_i = |\{\mathbf{c} \in \mathcal{C} \mid \mathrm{w}_H(\mathbf{c}) = i\}|$

$\mathcal{W}_{\mathcal{C}_1}(X) = \mathcal{W}_{\mathcal{C}_2}(X)$ but $\mathcal{C}_1 \not\sim \mathcal{C}_2$

Consider two binary $[6,3]$ codes $\mathcal{C}_1$ and $\mathcal{C}_2$ with respective generator matrices:

$$G_1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \text{ and } G_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

➜ Both codes have the same weight distribution: 1, 0, 3, 0, 3, 0, 1

➜ But they are not permutation-equivalent!

Punctured code

Let:

- ➜ $\mathcal{C}$ be an $[n, k]_q$ code

Punctured code

Let:

- ➜ $\mathcal{C}$ be an $[n, k]_q$ code
- ➜ $(J, \overline{J})$ be a partition of $\{1, \ldots, n\}$

Punctured code

Let:

- $\mathcal{C}$ be an $[n, k]_q$ code
- $(J, \overline{J})$ be a partition of $\{1, \dots, n\}$
- $\mathbf{x}_J$ the **restriction** of $\mathbf{x} \in \mathbb{F}_q^n$ to the coordinates indexed by J

Punctured code

Let:

- ➜ $\mathcal{C}$ be an $[n, k]_q$ code
- ➜ $(J, \overline{J})$ be a partition of $\{1, \ldots, n\}$
- ➜ $\mathbf{x}_J$ the **restriction** of $\mathbf{x} \in \mathbb{F}_q^n$ to the coordinates indexed by J

Punctured code $\mathcal{C}_J$

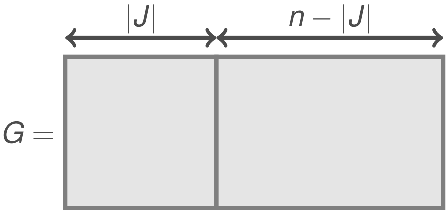
The words of $\mathcal{C}_J$ are codewords of $\mathcal{C}$ restricted to the $\overline{J}$-locations, i.e.

$$\mathcal{C}_J = \{\mathbf{c}_{\overline{J}} \mid \mathbf{c} \in \mathcal{C}\}$$

Punctured code

Let:

- ➜ $\mathcal{C}$ be an $[n, k]_q$ code
- ➜ $(J, \overline{J})$ be a partition of $\{1, \ldots, n\}$
- ➜ $\mathbf{x}_J$ the **restriction** of $\mathbf{x} \in \mathbb{F}_q^n$ to the coordinates indexed by J



Punctured code $\mathcal{C}_J$

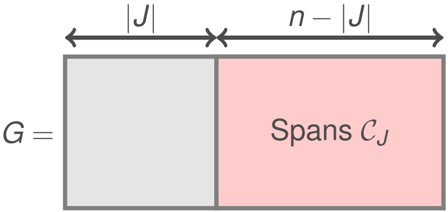
The words of $\mathcal{C}_J$ are codewords of $\mathcal{C}$ restricted to the $\overline{J}$-locations, i.e.

$$\mathcal{C}_J = \{\mathbf{c}_{\overline{J}} \mid \mathbf{c} \in \mathcal{C}\}$$

Punctured code

Let:

- $\mathcal{C}$ be an $[n, k]_q$ code
- $(J, \overline{J})$ be a partition of $\{1, \dots, n\}$
- $\mathbf{x}_J$ the **restriction** of $\mathbf{x} \in \mathbb{F}_q^n$ to the coordinates indexed by J



Punctured code $\mathcal{C}_J$

The words of $\mathcal{C}_J$ are codewords of $\mathcal{C}$ restricted to the $\overline{J}$-locations, i.e.

$$\mathcal{C}_J = \{\mathbf{c}_{\overline{J}} \mid \mathbf{c} \in \mathcal{C}\}$$

Signature

Signature

$\mathcal{S}$ is a **signature** if $\mathcal{S}(\mathcal{C}, i) = \mathcal{S}(\sigma(\mathcal{C}), \sigma(i))$

Signature

Signature

$\mathcal{S}$ is a **signature** if $\mathcal{S}(\mathcal{C}, i) = \mathcal{S}(\sigma(\mathcal{C}), \sigma(i))$

Building a signature from an invariant: If $\mathcal{V}$ is an invariant then,

Signature

Signature

$\mathcal{S}$ is a **signature** if $\mathcal{S}(\mathcal{C}, i) = \mathcal{S}(\sigma(\mathcal{C}), \sigma(i))$

Building a signature from an invariant: If $\mathcal{V}$ is an invariant then,

$$\mathcal{C} \sim \hat{\mathcal{C}} \implies \begin{cases} \mathcal{V}(\mathcal{C}) = \mathcal{V}(\hat{\mathcal{C}}) \\ \end{cases}$$

Signature

Signature

$\mathcal{S}$ is a **signature** if $\mathcal{S}(\mathcal{C}, i) = \mathcal{S}(\sigma(\mathcal{C}), \sigma(i))$

Building a signature from an invariant: If $\mathcal{V}$ is an invariant then,

$$\mathcal{C} \sim \hat{\mathcal{C}} \implies \begin{cases} \mathcal{V}(\mathcal{C}) = \mathcal{V}(\hat{\mathcal{C}}) \\ \{\mathcal{V}(\mathcal{C}_i) \mid i \in \{1, \ldots, n\}\} = \{\mathcal{V}(\hat{\mathcal{C}}_i) \mid i \in \{1, \ldots, n\}\} \end{cases}$$

Where $\mathcal{C}_i$ is the punctured code $\mathcal{C}$ on i

Fully Discriminant Signatures

Fully Discriminant Signatures

A signature $\mathcal{S}$ is **fully discriminant** for $\mathcal{C}$ if:

$$\mathcal{S}(\mathcal{C}, i) \neq \mathcal{S}(\mathcal{C}, j) \text{ for all } i \neq j$$

Fully Discriminant Signatures

Fully Discriminant Signatures

A signature $\mathcal{S}$ is **fully discriminant** for $\mathcal{C}$ if:

$$\mathcal{S}(\mathcal{C}, i) \neq \mathcal{S}(\mathcal{C}, j) \text{ for all } i \neq j$$

How to retrieve the permutation? Suppose that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$

Fully Discriminant Signatures

Fully Discriminant Signatures

A signature $\mathcal{S}$ is **fully discriminant** for $\mathcal{C}$ if:

$$\mathcal{S}(\mathcal{C}, i) \neq \mathcal{S}(\mathcal{C}, j) \text{ for all } i \neq j$$

How to retrieve the permutation? Suppose that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$
If $\mathcal{S}$ is **fully discriminant** for $\mathcal{C}$ then:

$$\forall i \in \{1, \ldots, n\}, \exists \text{ unique } j \text{ such that } \mathcal{S}(\mathcal{C}_1, i) = \mathcal{S}(\mathcal{C}_2, j)$$

Fully Discriminant Signatures

Fully Discriminant Signatures

A signature $\mathcal{S}$ is **fully discriminant** for $\mathcal{C}$ if:

$$\mathcal{S}(\mathcal{C}, i) \neq \mathcal{S}(\mathcal{C}, j) \text{ for all } i \neq j$$

How to retrieve the permutation? Suppose that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$
If $\mathcal{S}$ is **fully discriminant** for $\mathcal{C}$ then:

$$\forall i \in \{1, \ldots, n\}, \exists \text{ unique } j \text{ such that } \mathcal{S}(\mathcal{C}_1, i) = \mathcal{S}(\mathcal{C}_2, j) \Longrightarrow \boxed{\sigma(i) = j}$$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

$$\begin{cases} \mathcal{C}_1 = \{110, 111, 010\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_1} = X + X^2 + X^3 \\ \mathcal{C}_2 = \{110, 011\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_2} = 2X^2 \\ \mathcal{C}_3 = \{110, 011, 100\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_3} = X + 2X^2 \\ \mathcal{C}_4 = \{111, 011, 101\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_4} = 2X^2 + X^3 \end{cases}$$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

$$\begin{cases} \mathcal{C}_1 = \{110, 111, 010\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_1} = X + X^2 + X^3 \\ \mathcal{C}_2 = \{110, 011\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_2} = 2X^2 \\ \mathcal{C}_3 = \{110, 011, 100\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_3} = X + 2X^2 \\ \mathcal{C}_4 = \{111, 011, 101\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_4} = 2X^2 + X^3 \end{cases}$$

$$\begin{cases} \hat{\mathcal{C}}_1 = \{011, 101\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_1} = 2X^2 \\ \hat{\mathcal{C}}_2 = \{011, 111, 101\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_2} = 2X^2 + X^3 \\ \hat{\mathcal{C}}_3 = \{001, 101, 111\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_3} = X + X^2 + X^3 \\ \hat{\mathcal{C}}_4 = \{001, 101, 110\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_4} = X + 2X^2 \end{cases}$$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

$$\begin{cases} \mathcal{C}_1 = \{110, 111, 010\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_1} = X + X^2 + X^3 \\ \mathcal{C}_2 = \{110, 011\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_2} = 2X^2 \\ \mathcal{C}_3 = \{110, 011, 100\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_3} = X + 2X^2 \\ \mathcal{C}_4 = \{111, 011, 101\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_4} = 2X^2 + X^3 \end{cases}$$

$$\begin{cases} \hat{\mathcal{C}}_1 = \{011, 101\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_1} = 2X^2 \\ \hat{\mathcal{C}}_2 = \{011, 111, 101\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_2} = 2X^2 + X^3 \\ \hat{\mathcal{C}}_3 = \{001, 101, 111\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_3} = X + X^2 + X^3 \\ \hat{\mathcal{C}}_4 = \{001, 101, 110\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_4} = X + 2X^2 \end{cases}$$

Thus $\sigma(1) = 3$ $\sigma(2) = 1$ $\sigma(3) = 4$ and $\sigma(4) = 2$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

$$\begin{cases} \mathcal{C}_1 = \{110, 111, 010\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_1} = X + X^2 + X^3 \\ \mathcal{C}_2 = \{110, 011\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_2} = 2X^2 \\ \mathcal{C}_3 = \{110, 011, 100\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_3} = X + 2X^2 \\ \mathcal{C}_4 = \{111, 011, 101\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_4} = 2X^2 + X^3 \end{cases}$$

$$\begin{cases} \hat{\mathcal{C}}_1 = \{011, 101\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_1} = 2X^2 \\ \hat{\mathcal{C}}_2 = \{011, 111, 101\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_2} = 2X^2 + X^3 \\ \hat{\mathcal{C}}_3 = \{001, 101, 111\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_3} = X + X^2 + X^3 \\ \hat{\mathcal{C}}_4 = \{001, 101, 110\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_4} = X + 2X^2 \end{cases}$$

Thus $\sigma(1) = 3$ $\sigma(2) = 1$ $\sigma(3) = 4$ and $\sigma(4) = 2$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

$$\begin{cases} \mathcal{C}_1 = \{110, 111, 010\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_1} = X + X^2 + X^3 \\ \mathcal{C}_2 = \{110, 011\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_2} = 2X^2 \\ \mathcal{C}_3 = \{110, 011, 100\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_3} = X + 2X^2 \\ \mathcal{C}_4 = \{111, 011, 101\} & \longrightarrow \quad \mathcal{W}_{\mathcal{C}_4} = 2X^2 + X^3 \end{cases}$$

$$\begin{cases} \hat{\mathcal{C}}_1 = \{011, 101\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_1} = 2X^2 \\ \hat{\mathcal{C}}_2 = \{011, 111, 101\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_2} = 2X^2 + X^3 \\ \hat{\mathcal{C}}_3 = \{001, 101, 111\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_3} = X + X^2 + X^3 \\ \hat{\mathcal{C}}_4 = \{001, 101, 110\} & \longrightarrow \quad \mathcal{W}_{\hat{\mathcal{C}}_4} = X + 2X^2 \end{cases}$$

Thus $\sigma(1) = 3$ $\sigma(2) = 1$ $\sigma(3) = 4$ and $\sigma(4) = 2$

Fully Discriminant Signatures

An Example of Fully Discriminant Signature

Let $\mathcal{C} = \{1110, 0111, 1010\}$ and $\hat{\mathcal{C}} = \{0011, 1011, 1101\}$

$$\begin{cases} \mathcal{C}_1 = \{110, 111, 010\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_1} = X + X^2 + X^3 \\ \mathcal{C}_2 = \{110, 011\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_2} = 2X^2 \\ \mathcal{C}_3 = \{110, 011, 100\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_3} = X + 2X^2 \\ \mathcal{C}_4 = \{111, 011, 101\} & \longrightarrow & \mathcal{W}_{\mathcal{C}_4} = 2X^2 + X^3 \end{cases}$$

$$\begin{cases} \hat{\mathcal{C}}_1 = \{011, 101\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_1} = 2X^2 \\ \hat{\mathcal{C}}_2 = \{011, 111, 101\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_2} = 2X^2 + X^3 \\ \hat{\mathcal{C}}_3 = \{001, 101, 111\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_3} = X + X^2 + X^3 \\ \hat{\mathcal{C}}_4 = \{001, 101, 110\} & \longrightarrow & \mathcal{W}_{\hat{\mathcal{C}}_4} = X + 2X^2 \end{cases}$$

Thus $\sigma(1) = 3$ $\sigma(2) = 1$ $\sigma(3) = 4$ and $\sigma(4) = 2$

Refined Signatures

Refined Signature

Let $\mathcal{S}$ be a signature. Let J be a subset of $\{1, \ldots, n\}$ If $\mathcal{C} \sim \hat{\mathcal{C}} \implies \mathcal{C}_J \sim \hat{\mathcal{C}}_J$
Thus $\mathcal{S}(\mathcal{C}_J, i)$ and $\mathcal{S}(\hat{\mathcal{C}}_{\mathcal{J}}, i)$ give additional information.

Refined Signatures

Refined Signature

Let $\mathcal{S}$ be a signature. Let J be a subset of $\{1, \ldots, n\}$ If $\mathcal{C} \sim \hat{\mathcal{C}} \implies \mathcal{C}_J \sim \hat{\mathcal{C}}_J$
Thus $\mathcal{S}(\mathcal{C}_J, i)$ and $\mathcal{S}(\hat{\mathcal{C}}_{\mathcal{J}}, i)$ give additional information.

Example of Refined Signature

$$\mathcal{C} = \left\{ \begin{array}{c} 01101, 01011, \\ 01110, 10101, 11110 \end{array} \right\} \quad \text{and} \quad \hat{\mathcal{C}} = \left\{ \begin{array}{c} 10101, 00111, \\ 10011, 11100, 11011 \end{array} \right\}$$

Refined Signatures

Refined Signature

Let $\mathcal{S}$ be a signature. Let J be a subset of $\{1, \ldots, n\}$ If $\mathcal{C} \sim \hat{\mathcal{C}} \implies \mathcal{C}_J \sim \hat{\mathcal{C}}_J$
Thus $\mathcal{S}(\mathcal{C}_J, i)$ and $\mathcal{S}(\hat{\mathcal{C}}_{\mathcal{J}}, i)$ give additional information.

Example of Refined Signature

$$\mathcal{C} = \left\{ \begin{array}{c} 01101, 01011, \\ 01110, 10101, 11110 \end{array} \right\} \quad \text{and} \quad \hat{\mathcal{C}} = \left\{ \begin{array}{c} 10101, 00111, \\ 10011, 11100, 11011 \end{array} \right\}$$

$$\begin{array}{lcl} \mathcal{W}_{\mathcal{C}_1}(X) = \mathcal{W}_{\hat{\mathcal{C}}_2}(X) & \implies & \sigma(1) = 2 \\ \mathcal{W}_{\mathcal{C}_4}(X) = \mathcal{W}_{\hat{\mathcal{C}}_4}(X) & \implies & \sigma(4) = 4 \\ \mathcal{W}_{\mathcal{C}_5}(X) = \mathcal{W}_{\hat{\mathcal{C}}_3}(X) & \implies & \sigma(5) = 3 \end{array}$$

Refined Signatures

Refined Signature

Let $\mathcal{S}$ be a signature. Let J be a subset of $\{1, \ldots, n\}$ If $\mathcal{C} \sim \hat{\mathcal{C}} \implies \mathcal{C}_J \sim \hat{\mathcal{C}}_J$
Thus $\mathcal{S}(\mathcal{C}_J, i)$ and $\mathcal{S}(\hat{\mathcal{C}}_{\mathcal{J}}, i)$ give additional information.

Example of Refined Signature

$$\mathcal{C} = \left\{ \begin{array}{c} 01101, 01011, \\ 01110, 10101, 11110 \end{array} \right\} \quad \text{and} \quad \hat{\mathcal{C}} = \left\{ \begin{array}{c} 10101, 00111, \\ 10011, 11100, 11011 \end{array} \right\}$$

Note that: $\mathcal{W}_{\mathcal{C}_2}(X) = \mathcal{W}_{\mathcal{C}_3}(X) = \mathcal{W}_{\hat{\mathcal{C}}_1}(X) = \mathcal{W}_{\hat{\mathcal{C}}_5}(X)$.
Thus: positions $\{2, 3\}$ in $\mathcal{C}$ and $\{1, 5\}$ in $\hat{\mathcal{C}}$ **cannot be discriminated**

Refined Signatures

Refined Signature

Let $\mathcal{S}$ be a signature. Let J be a subset of $\{1,\ldots,n\}$ If $\mathcal{C} \sim \hat{\mathcal{C}} \implies \mathcal{C}_J \sim \hat{\mathcal{C}}_J$
Thus $\mathcal{S}(\mathcal{C}_J, i)$ and $\mathcal{S}(\hat{\mathcal{C}}_{\mathcal{J}}, i)$ give additional information.

Example of Refined Signature

$$\mathcal{C} = \left\{ \begin{array}{c} 01101, 01011, \\ 01110, 10101, 11110 \end{array} \right\} \quad \text{and} \quad \hat{\mathcal{C}} = \left\{ \begin{array}{c} 10101, 00111, \\ 10011, 11100, 11011 \end{array} \right\}$$

Note that: $\mathcal{W}_{\mathcal{C}_2}(X) = \mathcal{W}_{\mathcal{C}_3}(X) = \mathcal{W}_{\hat{\mathcal{C}}_1}(X) = \mathcal{W}_{\hat{\mathcal{C}}_5}(X)$.
Thus: positions $\{2,3\}$ in $\mathcal{C}$ and $\{1,5\}$ in $\hat{\mathcal{C}}$ **cannot be discriminated**

$$\begin{array}{lll} \mathcal{W}_{\mathcal{C}_{\{1,2\}}} = \mathcal{W}_{\hat{\mathcal{C}}_{\{2,5\}}} & \implies \sigma(\{1,2\}) = \{2,5\} \\ \mathcal{W}_{\mathcal{C}_{\{1,3\}}} = \mathcal{W}_{\hat{\mathcal{C}}_{\{2,1\}}} & \implies \sigma(\{1,3\}) = \{2,1\} \end{array}$$

Thus $\sigma(2) = 5$ and $\sigma(3) = 1$

Refined Signatures

Refined Signature

Let $\mathcal{S}$ be a signature. Let J be a subset of $\{1, \ldots, n\}$ If $\mathcal{C} \sim \hat{\mathcal{C}} \implies \mathcal{C}_J \sim \hat{\mathcal{C}}_J$
Thus $\mathcal{S}(\mathcal{C}_J, i)$ and $\mathcal{S}(\hat{\mathcal{C}}_{\mathcal{J}}, i)$ give additional information.

Example of Refined Signature

$$\mathcal{C} = \left\{ \begin{array}{c} 01101, 01011, \\ 01110, 10101, 11110 \end{array} \right\} \quad \text{and} \quad \hat{\mathcal{C}} = \left\{ \begin{array}{c} 10101, 00111, \\ 10011, 11100, 11011 \end{array} \right\}$$

Note that: $\mathcal{W}_{\mathcal{C}_2}(X) = \mathcal{W}_{\mathcal{C}_3}(X) = \mathcal{W}_{\hat{\mathcal{C}}_1}(X) = \mathcal{W}_{\hat{\mathcal{C}}_5}(X)$.
Thus: positions $\{2, 3\}$ in $\mathcal{C}$ and $\{1, 5\}$ in $\hat{\mathcal{C}}$ **cannot be discriminated**

$$\begin{array}{lll} \mathcal{W}_{\mathcal{C}_{\{1,2\}}} = \mathcal{W}_{\hat{\mathcal{C}}_{\{2,5\}}} & \implies \sigma(\{1, 2\}) = \{2, 5\} \\ \mathcal{W}_{\mathcal{C}_{\{1,3\}}} = \mathcal{W}_{\hat{\mathcal{C}}_{\{2,1\}}} & \implies \sigma(\{1, 3\}) = \{2, 1\} \end{array}$$

Thus $\sigma(2) = 5$ and $\sigma(3) = 1$

Some notation

From now on, let $\mathcal{C}$ be a linear code of length n defined over $\mathbb{F}_q$. We denote

- Its **dimension** by $K(\mathcal{C})$.
- Its **minimum distance** by $d(\mathcal{C})$.

Support Splitting Algorithm

The Algorithm:

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Support Splitting Algorithm

The Algorithm:

Input: A signature $\mathcal{S}$ and two codes: $\mathcal{C}_1$ and $\mathcal{C}_2$.

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Support Splitting Algorithm

The Algorithm:

Input: A signature $\mathcal{S}$ and two codes: $\mathcal{C}_1$ and $\mathcal{C}_2$.

1. Construct a sequence of signatures:

$$\mathcal{S}_0 = \mathcal{S}, \mathcal{S}_1, \ldots, \mathcal{S}_r$$

of increasing *“discriminancy”* such that $\mathcal{S}_r$ is **fully discriminant** for $\mathcal{C}$.

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Support Splitting Algorithm

The Algorithm:

Input: A signature $\mathcal{S}$ and two codes: $\mathcal{C}_1$ and $\mathcal{C}_2$.

1. Construct a sequence of signatures:

$$\mathcal{S}_0 = \mathcal{S}, \mathcal{S}_1, \ldots, \mathcal{S}_r$$

   of increasing *"discriminancy"* such that $\mathcal{S}_r$ is **fully discriminant** for $\mathcal{C}$.
2. From $\mathcal{S}_r$ we retrieve σ such that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Support Splitting Algorithm

The Algorithm:

Input: A signature $\mathcal{S}$ and two codes: $\mathcal{C}_1$ and $\mathcal{C}_2$.

1. Construct a sequence of signatures:

$$\mathcal{S}_0 = \mathcal{S}, \mathcal{S}_1, \ldots, \mathcal{S}_r$$

   of increasing *"discriminancy"* such that $\mathcal{S}_r$ is **fully discriminant** for $\mathcal{C}$.
2. From $\mathcal{S}_r$ we retrieve σ such that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$

Proposal of signature: $\mathcal{S}(\mathcal{C}, i) = \mathcal{W}_{\mathcal{H}(\mathcal{C}_i)}(X)$ where $\mathcal{H}(\mathcal{C}) = \mathcal{C} \cap \mathcal{C}^{\perp}$

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Support Splitting Algorithm

The Algorithm:

Input: A signature $\mathcal{S}$ and two codes: $\mathcal{C}_1$ and $\mathcal{C}_2$.

1. Construct a sequence of signatures:

$$\mathcal{S}_0 = \mathcal{S}, \mathcal{S}_1, \ldots, \mathcal{S}_r$$

   of increasing *"discriminancy"* such that $\mathcal{S}_r$ is **fully discriminant** for $\mathcal{C}$.

2. From $\mathcal{S}_r$ we retrieve σ such that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$

Proposal of signature: $\mathcal{S}(\mathcal{C}, i) = \mathcal{W}_{\mathcal{H}(\mathcal{C}_i)}(X)$ where $\mathcal{H}(\mathcal{C}) = \mathcal{C} \cap \mathcal{C}^{\perp}$

- For **binary** codes $\mathcal{C}$ of length n and $h = \dim(\mathcal{H}(\mathcal{C}))$.
  The **(heuristic) complexity:** $\mathcal{O}\left(n^3 + 2^h n^2 \log(n)\right)$

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Support Splitting Algorithm

The Algorithm:

Input: A signature $\mathcal{S}$ and two codes: $\mathcal{C}_1$ and $\mathcal{C}_2$.

1. Construct a sequence of signatures:

$$\mathcal{S}_0 = \mathcal{S}, \mathcal{S}_1, \ldots, \mathcal{S}_r$$

   of increasing *"discriminancy"* such that $\mathcal{S}_r$ is **fully discriminant** for $\mathcal{C}$.
2. From $\mathcal{S}_r$ we retrieve σ such that $\mathcal{C}_2 = \sigma(\mathcal{C}_1)$

Proposal of signature: $\mathcal{S}(\mathcal{C}, i) = \mathcal{W}_{\mathcal{H}(\mathcal{C}_i)}(X)$ where $\mathcal{H}(\mathcal{C}) = \mathcal{C} \cap \mathcal{C}^{\perp}$

- For **binary** codes $\mathcal{C}$ of length n and $h = \dim(\mathcal{H}(\mathcal{C}))$.
  The **(heuristic) complexity:** $\mathcal{O}\left(n^3 + 2^h n^2 \log(n)\right)$
- When $h \longrightarrow 0$, Then the **Algorithm** runs in polynomial time.

N. Sendrier,
Finding the permutation between equivalent linear codes: The Support Splitting Algorithm,
IEEE Trans. on Inf. Theory, vol. 46(4), 2000.

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

Goppa code

Let:

- ➜ $L = (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_{2^m}$ with $\alpha_i \neq \alpha_j$ for all $i \neq j$.
- ➜ $g(X) \in \mathbb{F}_{2^m}[X]$ monic separable polynomial with $\deg(g) = t$ and $g(\alpha_i) \neq 0 \forall i$

$$\Gamma(L, g) = \mathrm{Alt}_t(\mathbf{a}, \mathbf{b}) = (\mathrm{GRS}_t(\mathbf{a}, \mathbf{b})) \cap \mathbb{F}_q$$

with $\mathbf{a} = L$ and $b_i = \dfrac{1}{g(a_i)}$

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

➜ A Goppa code $\mathcal{C} = \Gamma(L, g)$ has :

$$K(\mathcal{C}) \geq n - mt \quad \text{and} \quad d(\mathcal{C}) \geq t + 1$$

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

➜ A Goppa code $\mathcal{C} = \Gamma(L, g)$ has :

$$K(\mathcal{C}) \geq n - mt \quad \text{and} \quad d(\mathcal{C}) \geq t + 1$$

➜ Let $G_{\text{pub}} \in \mathbb{F}_2^{k \times n}$ be the public key of the McEliece scheme.

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

➜ A Goppa code $\mathcal{C} = \Gamma(L, g)$ has :

$$K(\mathcal{C}) \geq n - mt \quad \text{and} \quad d(\mathcal{C}) \geq t + 1$$

➜ Let $G_{\text{pub}} \in \mathbb{F}_2^{k \times n}$ be the public key of the McEliece scheme.

1. We enumerate all polynomials g of degree t over $\mathbb{F}_2^m$ such that $k \geq n - mt$.

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

➜ A Goppa code $\mathcal{C} = \Gamma(L, g)$ has :

$$K(\mathcal{C}) \geq n - mt \quad \text{and} \quad d(\mathcal{C}) \geq t + 1$$

➜ Let $G_{\text{pub}} \in \mathbb{F}_2^{k \times n}$ be the public key of the McEliece scheme.

1. We enumerate all polynomials g of degree t over $\mathbb{F}_2^m$ such that $k \geq n - mt$.
2. We check the equivalence with the **public code**.

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

➜ A Goppa code $\mathcal{C} = \Gamma(L, g)$ has :

$$K(\mathcal{C}) \geq n - mt \quad \text{and} \quad d(\mathcal{C}) \geq t + 1$$

➜ Let $G_{\text{pub}} \in \mathbb{F}_2^{k \times n}$ be the public key of the McEliece scheme.

1. We enumerate all polynomials g of degree t over $\mathbb{F}_2^m$ such that $k \geq n - mt$.
2. We check the equivalence with the **public code.**

There are $2^{498.55}$ binary Goppa codes!!
(for $n = 1024$ and $t = 50$)

Application in Code-Based Cryptography

The public key of the original McEliece scheme is a **randomly permuted binary Goppa code**.

➜ A Goppa code $\mathcal{C} = \Gamma(L, g)$ has :

$$K(\mathcal{C}) \geq n - mt \quad \text{and} \quad d(\mathcal{C}) \geq t + 1$$

➜ Let $G_{\text{pub}} \in \mathbb{F}_2^{k \times n}$ be the public key of the McEliece scheme.

1. We enumerate all polynomials g of degree t over $\mathbb{F}_2^m$ such that $k \geq n - mt$.
2. We check the equivalence with the **public code.**

There are $2^{498.55}$ binary Goppa codes!!
(for $n = 1024$ and $t = 50$)

Is necessary to use a **large** family of codes
to make this attack ineffective

4. Key Attacks

1. Introduction
2. Support Splitting Algorithm
3. **Distinguisher for GRS codes**
4. Attack against subcodes of GRS codes
5. Error-Correcting Pairs
6. Attack against GRS codes
7. Attack against Reed-Muller codes
8. Attack against Algebraic Geometry codes
9. Goppa codes still resist