The Support Splitting Algorithm and its Application to Code-based Cryptography

Dimitris E. Simos
(joint work with Nicolas Sendrier)

Project-Team SECRET
INRIA Paris-Rocquencourt

May 9, 2012
3rd Code-based Cryptography Workshop
Technical University of Denmark
Lyngby, Denmark
Outline of the Talk

Support Splitting Algorithm
  Mechanics
  Examples

McEliece Cryptosystem

Research Problems
Outline of the Talk

Support Splitting Algorithm
  Mechanics
  Examples

Applications
  McEliece Cryptosystem
  Research Problems
Code Equivalence
of Binary Codes

**Code Equivalence** Problem

- Two linear codes $C$ and $C'$ of length $n$ are (permutation)-equivalent if for some permutation $\sigma$ of $I_n = \{1, \ldots, n\}$ we have:
  \[ C' = \sigma(C) = \{(x_{\sigma^{-1}(i)})_{i \in I_n} \mid (x_i)_{i \in I_n} \in C\} \]

**Notation:** $C \sim C'$.

- Given two linear codes $C$ and $C'$, do we have $C \sim C'$?

**Motivation**

**Code equivalence** is difficult to decide:

1. not NP-complete
2. at least as hard as **Graph Isomorphism**

**Reference:** Petrank and Roth, IEEE-IT, 1997

**Goal**

Given two linear codes $C \sim C'$, find $\sigma$ such that $C' = \sigma(C)$
Invariants and Signatures
for a given Linear Code

Invariants of a Code

- A mapping $V$ is an invariant if $C \sim C' \Rightarrow V(C) = V(C')$
- Any invariant is a global property of a code

Weight Enumerators are Invariants

$C \sim C' \Rightarrow W_C(X) = W_{C'}(X)$ or $W_C(X) \neq W_{C'}(X) \Rightarrow C \not\sim C'$

- $W_C(X) = \sum_{i=0}^{n} A_i X^i$ and $A_i = |\{c \in C \mid w(c) = i\}|$

Signature of a Code

- A mapping $S$ is a signature if $S(\sigma(C), \sigma(i)) = S(C, i)$
- Property of the code and one of its positions (local property)

Building a Signature from an Invariant

1. If $V$ is an invariant, then $S_V : (C, i) \mapsto V(C_{\{i\}})$ is a signature
2. where $C_{\{i\}}$ is obtained by puncturing the code $C$ on $i$
3. If $C' = \sigma(C) \Rightarrow V(C_{\{i\}}) = V(C'_{\{\sigma(i)\}})$, $\forall i \in I_n$, i.e. $V = W$
The Support Splitting Algorithm (I)

Design of the Algorithm

Discriminant Signatures

1. A signature $S$ is discriminant for $C$ if $\exists i \neq j, S(C, i) \neq S(C, j)$
2. $S$ is fully discriminant for $C$ if $\forall i \neq j, S(C, i) \neq S(C, j)$

The Procedure

- From a given signature $S$ and a given code $C$, we wish to build a sequence $S_0 = S, S_1, \ldots, S_r$ of signatures of increasing “discriminancy” such that $S_r$ is fully discriminant for $C$
- Achieved by successive refinements of the signature $S$
- Reference: Sendrier, IEEE-IT, 2000

Statement

1. $SSA(C)$ returns a labeled partition $\mathcal{P}(S, C)$ of $I_n$
2. Assuming the existence of a fully discriminant signature, $SSA(C)$ recovers the desired permutation $\sigma$ of $C' = \sigma(C)$
An Example of a Fully Discriminant Signature

Statement
If $C' = \sigma(C)$ and $S$ is fully discriminant for $C$ then $\forall \ i \in I_n$ $\exists$ unique $j \in I_n$ such that $S(C, i) = S(C', j)$ and $\sigma(i) = j$

The Example

$C = \{1110, 0111, 1010\}$ and $C' = \{0011, 1011, 1101\}$

\[
\begin{align*}
C_{\{1\}} &= \{110, 111, 010\} & \rightarrow & \mathcal{W}_{C_{\{1\}}} (X) = X + X^2 + X^3 \\
C_{\{2\}} &= \{110, 011\} & \rightarrow & \mathcal{W}_{C_{\{2\}}} (X) = 2X^2 \\
C_{\{3\}} &= \{110, 011, 100\} & \rightarrow & \mathcal{W}_{C_{\{3\}}} (X) = X + 2X^2 \\
C_{\{4\}} &= \{111, 011, 101\} & \rightarrow & \mathcal{W}_{C_{\{4\}}} (X) = 2X^2 + X^3 \\
C'_{\{1\}} &= \{011, 101\} & \rightarrow & \mathcal{W}_{C'_{\{1\}}} (X) = 2X^2 \\
C'_{\{2\}} &= \{011, 111, 101\} & \rightarrow & \mathcal{W}_{C'_{\{2\}}} (X) = 2X^2 + X^3 \\
C'_{\{3\}} &= \{001, 101, 111\} & \rightarrow & \mathcal{W}_{C'_{\{3\}}} (X) = X + X^2 + X^3 \\
C'_{\{4\}} &= \{001, 101, 110\} & \rightarrow & \mathcal{W}_{C'_{\{4\}}} (X) = X + 2X^2
\end{align*}
\]

$C' = \sigma(C)$ where $\sigma(1) = 3$, $\sigma(2) = 1$, $\sigma(3) = 4$ and $\sigma(4) = 2$
An Example of a Refined Signature

The Example

\[ C = \{01101, 01011, 01110, 10101, 11110\} \]
\[ C' = \{10101, 00111, 10011, 11100, 11011\} \]

\[
\begin{align*}
\mathcal{W}_{C\{1\}}(X) &= X^2 + 3X^3 &= \mathcal{W}_{C'\{2\}}(X) \implies \sigma(1) = 2 \\
\mathcal{W}_{C\{4\}}(X) &= 2X^2 + 3X^3 &= \mathcal{W}_{C'\{4\}}(X) \implies \sigma(4) = 4 \\
\mathcal{W}_{C\{5\}}(X) &= 3X^2 + X^3 + X^4 &= \mathcal{W}_{C'\{3\}}(X) \implies \sigma(5) = 3 \\
\mathcal{W}_{C\{2\}}(X) &= 3X^2 + 2X^3 &= \mathcal{W}_{C'\{1\}}(X) \\
\mathcal{W}_{C\{3\}}(X) &= 3X^2 + 2X^3 &= \mathcal{W}_{C'\{5\}}(X)
\end{align*}
\]

Refinement: Positions \{2, 3\} in \(C\) and \{1, 5\} in \(C'\) cannot be discriminated, but

\[
\begin{align*}
\mathcal{W}_{C\{1,2\}}(X) &= 3X^2 &= \mathcal{W}_{C'\{2,5\}}(X) \implies \sigma(\{1, 2\}) = \{2, 5\} \\
\mathcal{W}_{C\{1,3\}}(X) &= X + 2X^2 + X^3 &= \mathcal{W}_{C'\{2,1\}}(X) \implies \sigma(\{1, 3\}) = \{2, 1\}
\end{align*}
\]

Thus \(\sigma(1) = 2, \sigma(2) = 5, \sigma(3) = 1, \sigma(4) = 4\) and \(\sigma(5) = 3\)

Fundamental Properties of \(SSA\)

1. If \(C' = \sigma(C)\) then \(\mathcal{P}'(S, C') = \sigma(\mathcal{P}(S, C))\)
2. The output of \(SSA(C)\) where \(C = \langle G \rangle\) is independent of \(G\)
The Support Splitting Algorithm (II)
Practical Issues

A Good Signature
The mapping $(C, i) \mapsto W_{H(C_i)}(X)$ where $H(C) = C \cap C^\perp$ is a signature which is, for random codes,
- easy to compute because of the small dimension (Sendrier, 1997)
- discriminant, i.e. $W_{H(C_i)}(X)$ and $W_{H(C_j)}(X)$ are “often” different

Algorithmic Cost
Let $C$ be a binary code of length $n$, and let $h = \dim(H(C))$:
- First step: $O(n^3) + O(n2^h)$
- Each refinement: $O(hn^2) + O(n2^h)$
- Number of refinements: $\approx \log n$
Total (heuristic) complexity: $O(n^3 + 2^h n^2 \log n)$

Implementation
Currently developed on GAP and MAGMA
Structural Attacks on McEliece-like Cryptosystems

Binary Goppa Code

Let \( L = \{\alpha_1, \ldots, \alpha_n\} \subset GF(2^m) \) and \( g(z) \in GF(2^m)[z] \) square-free of degree \( t \) with \( g(\alpha_i) \neq 0 \).

\[ \Gamma(L, g) = \{(c_1, \ldots, c_n) \in GF(2^m) \mid \sum_{i=1}^{n} \frac{c_i}{z-\alpha_i} \equiv 0 \mod g(z) \} \]

McEliece and Niederreiter Cryptosystems

- \( \Gamma \) a \( t \)-error correcting binary Goppa code

<table>
<thead>
<tr>
<th></th>
<th>McEliece</th>
<th>Niederreiter</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret key</td>
<td>gen. matrix ( G_0 ) of ( \Gamma )</td>
<td>parity check matrix ( H_0 ) of ( \Gamma )</td>
</tr>
<tr>
<td></td>
<td>permutation matrix ( P )</td>
<td>permutation matrix ( P )</td>
</tr>
<tr>
<td>public key</td>
<td>( G = SG_0P )</td>
<td>( H = UH_0P )</td>
</tr>
</tbody>
</table>

Attacking McEliece Cryptosystem with SSA

1. Enumeration of all polynomial \( g \) of a family \( G \) of \( \Gamma(L, g) \) and check equivalence with the public code
2. There are \( 2^{498.55} \) \((m = 1024, t = 524)\) binary Goppa codes!
Weak Keys in the McEliece Cryptosystem

Weak Keys
Binary Goppa codes with binary generator polynomials $g$

Detection of Weak Keys with $SSA$

1. Compute $SSA(C) = \mathcal{P}(S, C)$ where $C$ is the public code
2. If the cardinalities of the cells of $\mathcal{P}$ are equal to the cardinalities of the conjugacy cosets of $L$ then $C \sim \Gamma(L, g)$ where $g$ has binary coefficients (with a high probability)

Enumerative Attack with $SSA$

1. For all binary polynomial $g$ of given degree $t$ compute $SSA(\Gamma(L, g)) = \mathcal{P}'(S, \Gamma(L, g))$
2. If $\mathcal{P}'(S, \Gamma(L, g)) \sim \mathcal{P}(S, C)$ then return $g$
3. Efficient for $\Gamma(L, g)$ of length 1024 with $g$ of degree 50 using idempotent subcodes (Loidreau and Sendrier, IEEE-IT, 2001)
Research Problems
Related to Coding Theory

**Code Equivalence** over $GF(q)$, $q > 2$

Two linear codes $C$ and $C'$ of length $n$ are equivalent over $GF(q)$ if $C'$ can be obtained from $C$ by a series of transformations:

1. Permutation of the codeword positions
2. Multiplication in a position by non-zero elements of $GF(q)$
3. Application of field automorphism to all codeword positions

Research Problem
Given $C$ and $C'$ decide $C \sim C'$ over $GF(q)$?

Current Approach
Generalized SSA:

1. Codes with non-trivial automorphism groups
2. Codes with large hulls (i.e., self-dual, $C = C^\perp$)
3. ...
Research Problems
Related to Code-based Cryptography

Research Problem
Measure the key security of code-based cryptosystems over $GF(q)$

Wild McEliece Cryptosystem
Proposed by Bernstein, Lange and Peters, SAC, 2010
- Uses wild Goppa codes ($g$ is in $F_{q^m}[x]$)
- Estimation of the key security with the generalized SSA?

Research Problem
Other structural attacks for code-based cryptosystems?

Detection of Weak Keys
Apply SSA for other (sub)-families of hidden codes
Summary

Highlights

1. We presented the basic concepts of the support splitting algorithm for solving the Code Equivalence problem for the binary case.

2. We showed a structural attack of SSA to code-based cryptosystems (McEliece, Niederreiter).
Summary

Highlights

1. We presented the basic concepts of the support splitting algorithm for solving the Code Equivalence problem for the binary case.
2. We showed a structural attack of SSA to code-based cryptosystems (McEliece, Niederreiter).

Future Work

Solve (some) of the research problems..!


Questions - Comments

Thanks for your Attention!