CFS Software Implementation

Gregory Landais   Nicolas Sendrier

INRIA Paris-Rocquencourt, Project-Team SECRET

May 9, 2012

CFS

First code-based signature scheme. Relies on :

- hardness of the syndrome decoding problem
- the undistinguishability of a binary Goppa code

Timeline :

2001 Publication by N. Courtois, M. Finiasz, N.Sendrier.
2004 FPGA implementation, signing time under 1 second.
200? Unpublished Bleichenbacher's attack.
2010 Parallel CFS.
2011 Distinguisher for low rate Goppa codes.

CFS instance

A CFS instance is defined by a binary Goppa code Γ :

- of length $n \leq 2^m$
- of support $L = (\alpha_0, \ldots, \alpha_{n-1})$, an ordered sequence of distincts elements of $\mathbb{F}_{2^m}$
- of polynomial generator g of degree t
- with an algebraic t-error correcting procedure
- of dimension $k \leq n - m \times t$
- of parity check matrix $H \in \{0, 1\}^{n \times (n-k)}$

Parameters : m, t

Public key : H

Secret key : L, g

CFS

```
function SIGN(M)                                ▷ input: message M
    S ← syndromes(M)                ▷ S is a family of syndromes
                                      (typically obtained by hashing)
    for all s ∈ S do
        e ← decode(s)
        if e ≠ fail then
            return e, s
        end if
    end for
end function
```

Probability of success of the decoding $\approx \frac{1}{t!}$

Let's open the black box

```
function SIGN(M)                              ▷ input: message M
    S ← syndromes(M)
    for all s ∈ S do
        σ(z) ← solve_key_eq(s)
        e ← roots(σ(z))
        if card(e) = t then
            return e, s
        end if
    end for
end function
```

Generating the family of syndromes

1. Counter appending : append a counter to the message before hashing it to a syndrome.
   - Hashing performed on the target architecture
   - Variable signature size
   - No Parallel-CFS counter measure

   BAD IDEA

2. Complete decoding : hash the message to a unique syndrome and try to guess δ elements of the corresponding error pattern.
   - Adds a recoverable signature failure probability

   BETTER IDEA

Loop body diet

```
function SIGN(M)                                  ▷ input: message M
    s_0 ← hash(M)
    for all e ∈ E do                              ▷ E is the set of error pattern of weight δ
        s ← s_0 + syndrome(e)
        σ(z) ← solve_key_eq(s)
        if σ(z) splits in F_{2^m}[z] then
            return roots(σ(z)), e
        end if
    end for
end function
```

Let's count

		critical				non critical	
(m, t)	type	(1)	(2)	(3)	(1)+(2)+(3)	(4)	(5)
(18,9)	BM	58	180	840	1078	2184	3079.1
(18,9)	Pat.	38	329	840	1207	1482	3079.1
(20,8)	BM	52	144	747	943	1950	3024.6
(20,8)	Pat.	34	258	747	1039	1326	3024.6

(1) syndrome adjustment
(2) key equation solving
(3) split checking
(4) initial syndrome
(5) root finding

Table: Number of field operations (excluding additions) per decoding

Finite field operations

Store logarithm and the exponentiation of each element in base α, a primitive element of $\mathbf{F}_{2^m}$.
Space used :

- $\mathbf{F}_{2^{20}}$ $2^{20} \times 2 \times 4\mathrm{B} = 8192\mathrm{KB}$
- $\mathbf{F}_{2^{10}}$ $2^{10} \times 2 \times 2\mathrm{B} = 4\mathrm{KB}$

Cache size of Intel XEON W3550 :

- L1 $128\mathrm{KB}$
- L2 $1024\mathrm{KB}$
- L3 $8192\mathrm{KB}$

Timings

	(m, t, w, λ)			
	(18,9,11,3)	(18,9,11,4)	(20,8,10,3)	(20,8,9,5)
decoding	1 117 008	1 489 344	121 262	360 216
BM	14.70 s	19.61 s	1.32 s	3.75 s
Pat	15.26 s	20.34 s	1.55 s	4.26 s
sec bits	83.4	87.0	82.5	87.3

Table: Average number of algebraic decoding and running time per signature

Conclusion

Signing with codes and 80 bits of security in less than 1 second is possible.

TODO list

- Make the code public
- Benchmark it (eBACS)
- Bit-slice it (joint work with Peter Schwabe)
- FPGA it (joint work with Jean-Luc Beuchat)

Thank you

Questions ?