5. Other Cryptographic Constructions Relying on Coding Theory

- Code-Based Digital Signatures
- The Courtois-Finiasz-Sendrier (CFS) Construction
- Attacks against the CFS Scheme
- **Parallel-CFS**
- Stern's Zero-Knowledge Identification Scheme
- An Efficient Provably Secure One-Way Function
- The Fast Syndrome-Based (FSB) Hash Function

The Idea of Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

The Idea of Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

For the signer:

- signature takes twice more computation
- the signature is twice longer

The Idea of Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

For the signer:

- signature takes twice more computation
- the signature is twice longer

For the attacker:

- he has to forge 2 signatures

The Idea of Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

For the signer:

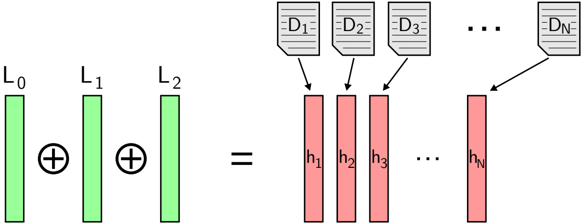
- signature takes twice more computation
- the signature is twice longer

For the attacker:

- he has to forge 2 signatures

Well, things are a little more complicated than that...

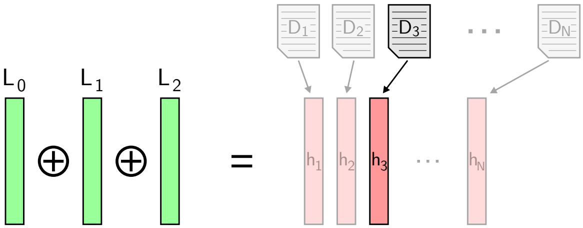
Decoding One Out of Many, Twice?



Start from a set of N documents:

- compute their hashes h_i to build a list

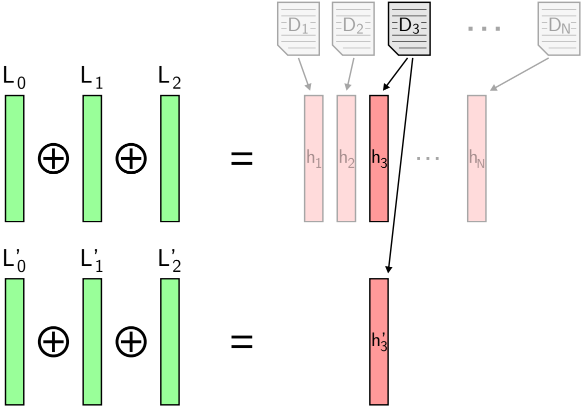
Decoding One Out of Many, Twice?



Start from a set of N documents:

- compute their hashes h_i to build a list
- when $N = 2^{\frac{mt}{3}}$, one solution is found

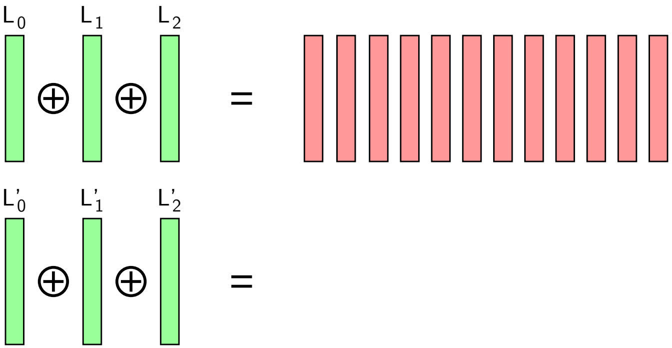
Decoding One Out of Many, Twice?



Then, move on to the second hash function *h'*:

- problem: there is only one target syndrome left
  → both signatures must be for the same document

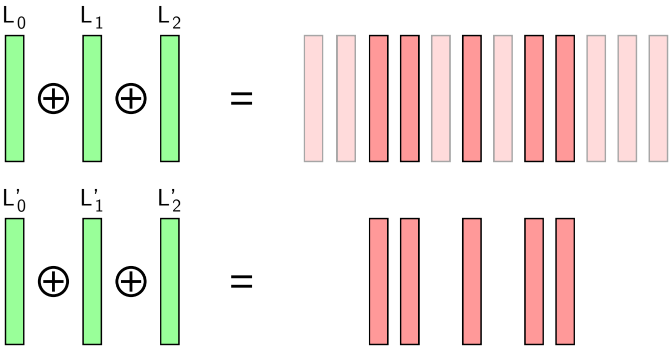
Decoding One Out of Many, Twice?



To chain one out of many attacks:

- build a larger set of syndromes

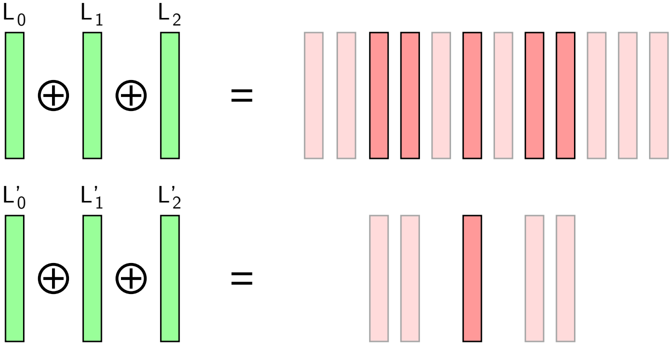
Decoding One Out of Many, Twice?



To chain one out of many attacks:

- build a larger set of syndromes
- find a (large) set of solutions

Decoding One Out of Many, Twice?



To chain one out of many attacks:

- build a larger set of syndromes
- find a (large) set of solutions
- use this set to find a “double” solution

Parallel-CFS Requires Complete Decoding

A similar problem happens to the legitimate signer when using counters.

A simple signing strategy would be to:

- pick a document D to sign
- use hash function h to compute a signature
  → this first signature uses a counter value i
- then, using h', compute a second signature, with counter value i'

Parallel-CFS Requires Complete Decoding

A similar problem happens to the legitimate signer when using counters.

A simple signing strategy would be to:

- pick a document D to sign
- use hash function h to compute a signature
  → this first signature uses a counter value i
- then, using h', compute a second signature, with counter value i'

Problem: the attacker can do the same

- pick a document D to sign
- build a list of syndromes using h and different counters
  → forge a first signature
- forge a second signature, using h' and a list of counters

Parallel-CFS Requires Complete Decoding

A similar problem happens to the legitimate signer when using counters.

A simple signing strategy would be to:

- pick a document D to sign
- use hash function h to compute a signature
  → this first signature uses a counter value i
- then, using h', compute a second signature, with counter value i'

For the Parallel-CFS construction to work:

- the input of both hash functions should be the same
- both signatures should use the same counter value
- both syndromes are decodable with probability $\left(\frac{1}{t!}\right)^2$

The complete decoding version is much more efficient!

Security Analysis of Parallel-CFS

With the complete decoding version of CFS, the size of lists L_0, L_1, and L_2 can be such that $|L_0| \times |L_1| \times |L_2| = 2^{mt}$.

Security Analysis of Parallel-CFS

With the complete decoding version of CFS, the size of lists L_0, L_1, and L_2 can be such that $|L_0| \times |L_1| \times |L_2| = 2^{mt}$.

Say the attacker wants to forge 2^c signatures with h, he can pick:

- $|L_S| = 2^{\frac{mt+2c}{3}}$, $|L_0| = |L_1| = 2^{\frac{mt+c/2}{3}}$, and $|L_2| = 2^{\frac{mt-c}{3}}$
- merge the lists pairwise, zeroing $\frac{mt-c}{3}$ bits
- obtain 2^c solutions on average

→ the cost of this step is $2^{\frac{mt+2c}{3}}$

Security Analysis of Parallel-CFS

With the complete decoding version of CFS, the size of lists L_0, L_1, and L_2 can be such that $|L_0| \times |L_1| \times |L_2| = 2^{mt}$.

Say the attacker wants to forge 2^c signatures with h, he can pick:

- $|L_S| = 2^{\frac{mt+2c}{3}}$, $|L_0| = |L_1| = 2^{\frac{mt+c/2}{3}}$, and $|L_2| = 2^{\frac{mt-c}{3}}$
- merge the lists pairwise, zeroing $\frac{mt-c}{3}$ bits
- obtain 2^c solutions on average

→ the cost of this step is $2^{\frac{mt+2c}{3}}$

Then, to forge 1 signature with h', the attacker uses:

- $|L_S| = 2^c$, $|L_0| = |L_1| = 2^{\frac{mt+c}{4}}$, and $|L_2| = 2^{\frac{mt-c}{2}}$
- merge the lists pairwise, zeroing c bits
- obtain 1 solution on average

→ the cost of this step is $2^{\frac{mt-c}{2}}$

Security Analysis of Parallel-CFS

Security of Parallel-CFS with 2 Signatures

The optimal choice of c is when $\frac{mt+2c}{3} = \frac{mt-c}{2}$, that is $c = \frac{1}{7}mt$.

This gives a total chained GBA attack cost of $2^{\frac{3}{7}mt}$.

Security Analysis of Parallel-CFS

Security of Parallel-CFS with 2 Signatures

The optimal choice of c is when $\frac{mt+2c}{3} = \frac{mt-c}{2}$, that is $c = \frac{1}{7}mt$.

This gives a total chained GBA attack cost of $2^{\frac{3}{7}mt}$.

Security of Parallel-CFS with i Signatures

When using i signatures in parallel, the cost of the attacks becomes $2^{\frac{2^i-1}{2^{i+1}-1}mt}$.

It can be made very close to $2^{\frac{mt}{2}}$: $\frac{1}{3}, \frac{3}{7}, \frac{7}{15}, \dots$

Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

For the signer:

- signature takes twice more computation
- the signature is twice longer

Parallel-CFS

What happens if you use 2 hash functions h and h' to compute 2 different CFS signatures?

For the signer:

- signature takes twice more computation
- the signature is twice longer

For the attacker:

- the cost of forgery is significantly increased!

Parallel-CFS allows to use smaller, more efficient, parameters than the original CFS.

5. Other Cryptographic Constructions Relying on Coding Theory

- Code-Based Digital Signatures
- The Courtois-Finiasz-Sendrier (CFS) Construction
- Attacks against the CFS Scheme
- Parallel-CFS
- **Stern's Zero-Knowledge Identification Scheme**
- An Efficient Provably Secure One-Way Function
- The Fast Syndrome-Based (FSB) Hash Function