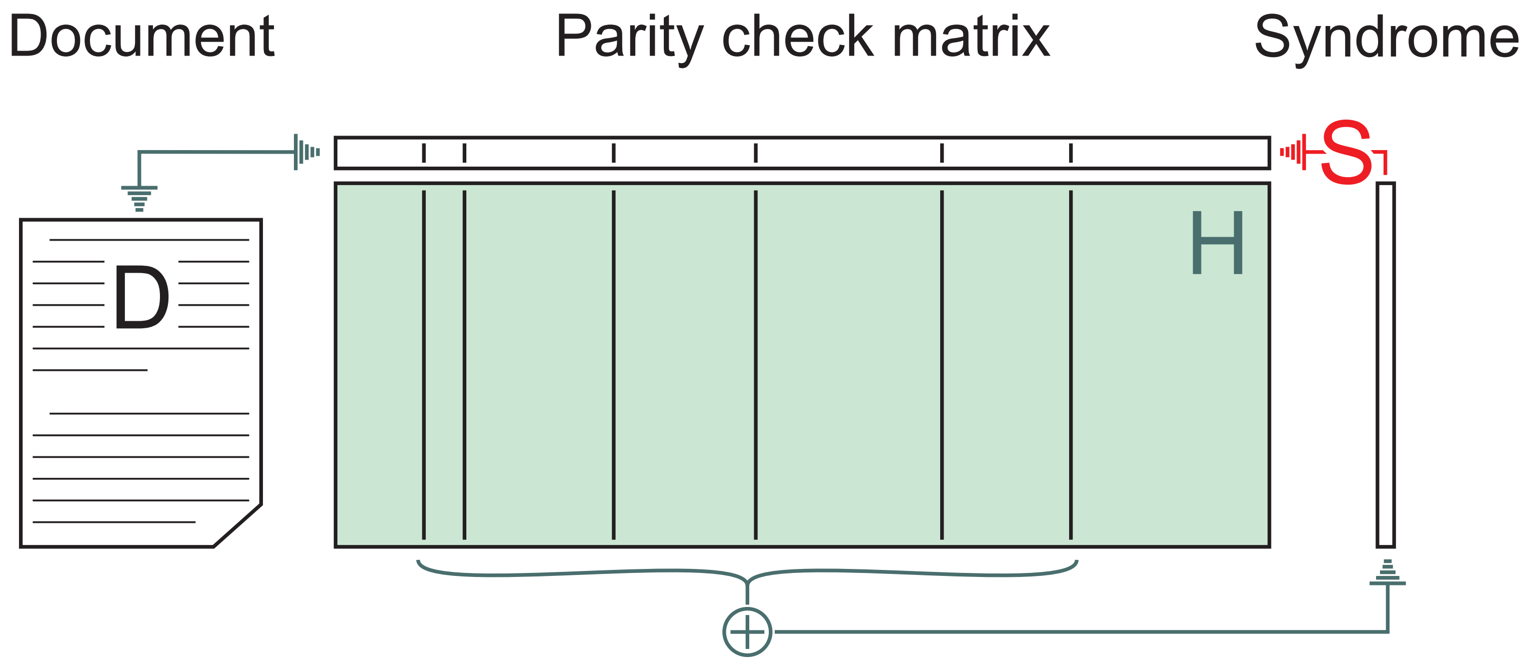
How to achieve a McEliece-based Digital Signature Scheme

Nicolas Courtois Matthieu Finiasz Nicolas Sendrier

ASIACRYPT 2001 – Brisbane

INRIA

McEliece in a nutshell
(Niederreiter version)

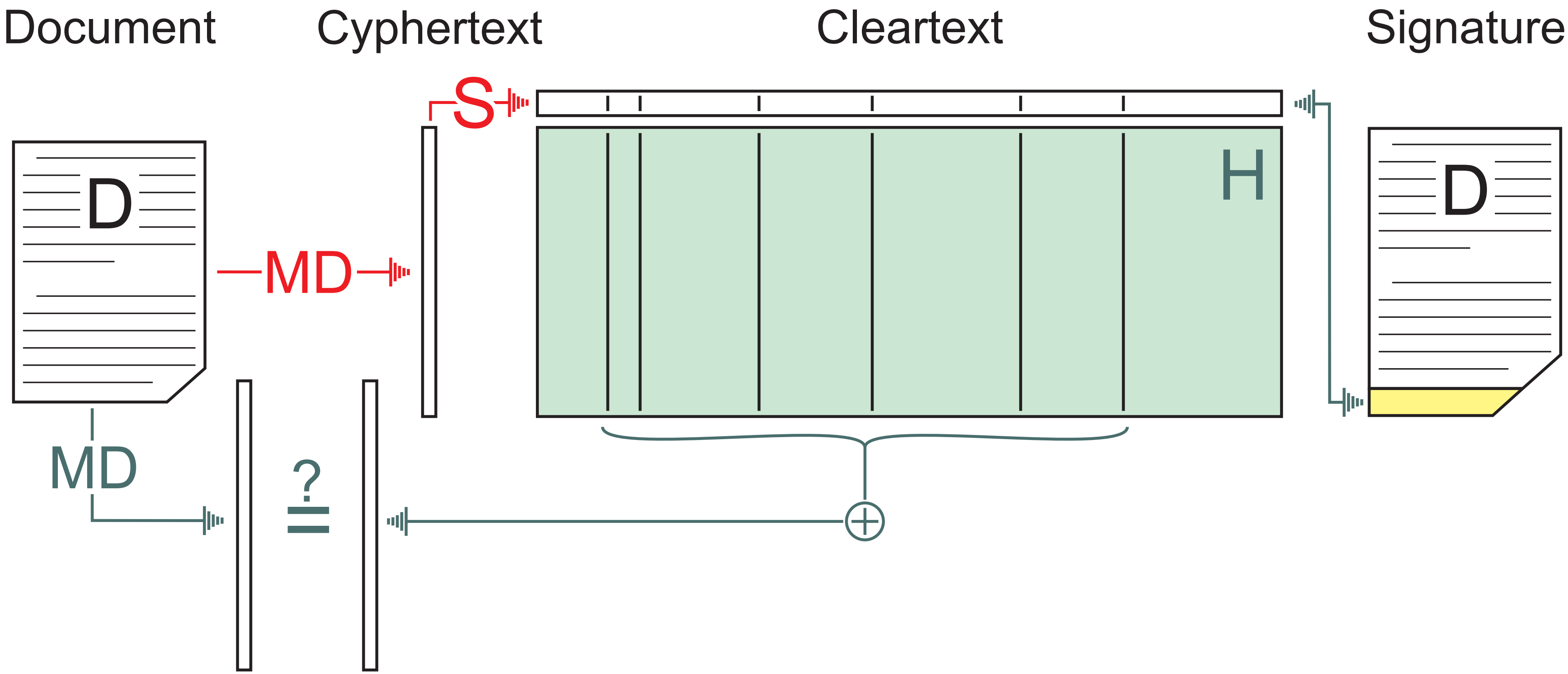


⇒ This scheme is equivalent to the original McEliece scheme, but is more practical.

From Public-key Cryptography to Digital Signature

⇒ A digital signature consists in adding a few bits to a file in order to prove both its origin and its content.

⇒ Any public key cryptosystem can be transformed in a signature scheme like this:



Using error-correcting codes. . .

To perform this with McEliece, one has to be able to decode any syndrome returned by the hash function.

⚠ Niederreiter coding is not a one to one mapping.
$\Rightarrow$ some syndromes are not the image of a message

With the original parameters: $t = 50$, $m = 10$, $n = 1024$.

- ◇ there are 2^{500} different syndromes (of length 500)
- ◇ there are $\binom{1024}{50} \simeq 2^{284}$ sums of 50 columns of H

$\Rightarrow$ This makes a ratio of 1 decodable syndrome out of 2^{216}.

> We need to:
>
> - ◇ find a way to decode any syndrome
> - ◇ or find a decodable syndrome related to the document

Solving this problem

$\Rightarrow$ we need to take advantage of the t-error decoding method

Find a way to decode more syndromes: decode syndromes corresponding to error patterns of greater weight

$\Rightarrow$ possible using exhaustive search

Find a decodable syndrome

$\Rightarrow$ Add a counter i to the document:

- $\diamond$ Hash the document and the counter at the same time: $[\cdots D \cdots][\cdot i \cdot] \longrightarrow h_i$
- $\diamond$ Try to decode each h_i until one is decodable
- $\diamond$ We denote i_0 the smallest index such that h_{i_0} is decodable

$\Rightarrow$ In both cases we need to change the parameters to obtain a better ratio.

INRIA

Better parameters

The ratio of decodable syndromes is easy to calculate:

$$\mathcal{R} = \frac{\mathcal{N}_{dec}}{\mathcal{N}_{tot}} = \frac{\binom{n}{t}}{2^n} \underset{n \text{ large}}{\simeq} \frac{1}{t!}$$

⇒ Hash document+counter $t!$ times in average to obtain a decodable syndrome

⚠ Telling if a syndrome is decodable is as hard as decoding it

⇒ We need to perform $t!$ decodings, each one having a complexity of $t^2(\log_2 n)^3$

> n only has a small influence: we will choose t to have a reasonnable signature time. t shouldn't be greater than 10, preferably 9.

INRIA

Secure parameters

We have a small t but still want a good security (about 2^{80} CPU operations)
$\Rightarrow$ n will be large

Number of binary operations for an attack

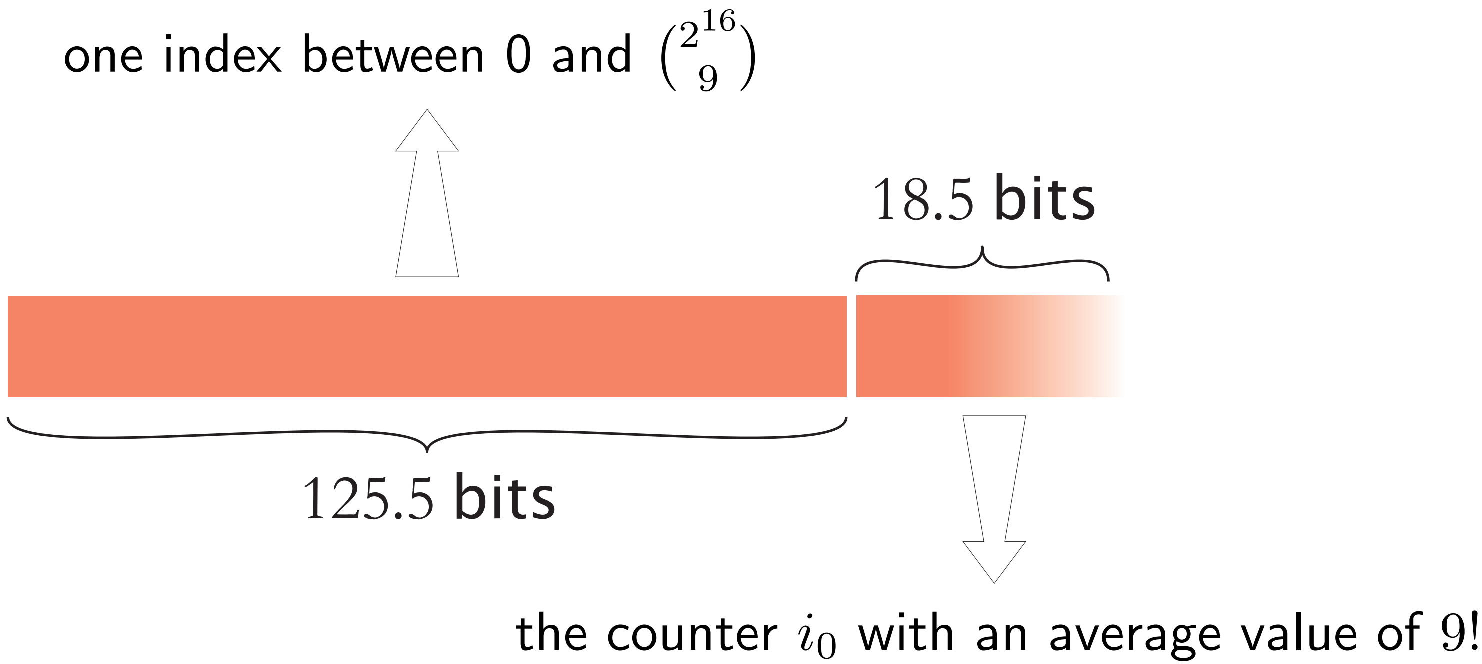
n	$t = 9$	$t = 10$
2^{13}	$2^{69.3}$	$2^{72.3}$
2^{14}	$2^{74.0}$	$2^{77.4}$
2^{15}	$2^{78.8}$	$2^{87.4}$
2^{16}	$2^{83.7}$	$2^{90.9}$
2^{17}	$2^{88.2}$	$2^{94.6}$

$$\begin{cases} t = 10 \text{ and } n = 15 \\ t = 9 \text{ and } n = 16 \end{cases} \quad \longleftarrow \text{10 times faster}$$

INRIA

Signature size

$\Rightarrow$ we index all the words of weight 9 and length 2^{16}.



$\Rightarrow$ The counter must be present for verification. It can be made of fixed length.

$\Rightarrow$ Signature is in average 144 bits long.

Reducing the signature size. . .

Verification is very fast (summing 9 columns of H and hashing one file)

⇒ The signature can be shortened by omitting some information: verificator will then try all possible values

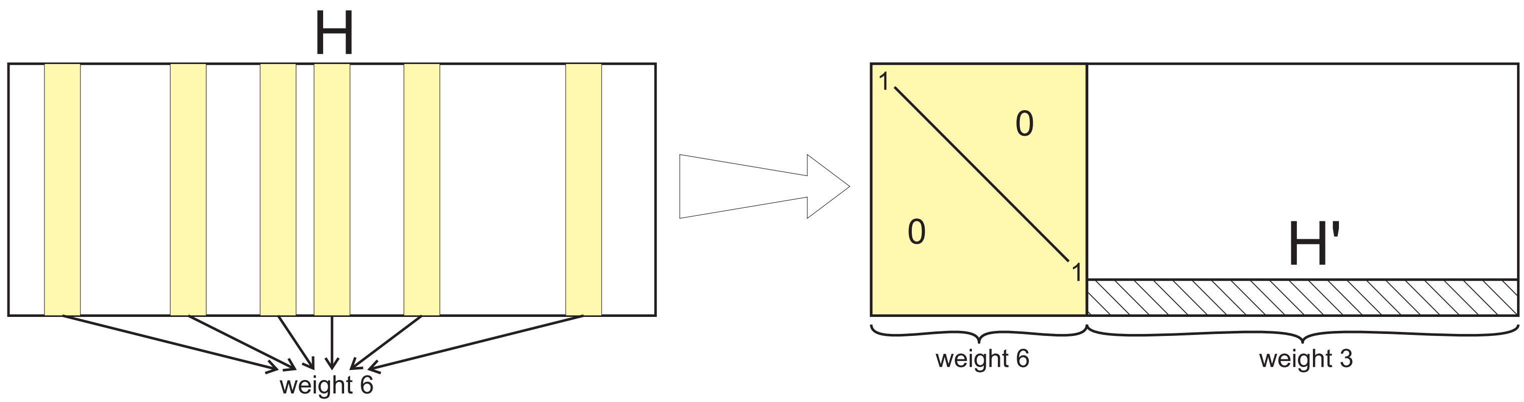
⇒ Signature will contain less than t positions

omitted positions	signature length		verification	
	partial	total	WF	time
0	125.5	144	9	$\sim \mu$s
1	112.7	131	9	$\sim \mu$s
2	99.7	118	2^{14}	$\sim$ ms
3	86.5	105	2^{27}	$\sim$ 30s
4	73.1	92	2^{40}	—
5	59.4	77	2^{54}	—

We can verify a signature of 105 bits in about 30 seconds.

Reducing more

We can reduce the signature size even more by giving only approximate positions
⇒ group the columns in small clusters of 16 columns



⇒ The matrix can easily be transformed with a Gaussian elimination (about 2^{24} column operations). We then have the same problem to solve.

⇒ We can get signatures of 81 bits.

Scalability

⇒ The signature algorithm is easily scalable. For one omitted position we have the following asymptotic values:

signature cost	$t!t^2m^3$
signature length	$(t-1)m + \log_2 t$
verification cost	t^2m
public key size	$tm2^m$
cost of best decoding attack	$2^{tm(1/2+o(1))}$

⇒ Security increases much faster than any other parameter

Conclusion

- Signature using McEliece is possible!
- The algorithm obtained is polymorphic. It gives:
  - either very short signatures of 81 bits
  - or short signatures $(131$ or 118 bits) with a faster verification
- the signature time is long (about 1 minute)
- the public key is large (1MB)
- its security relies on well known hard problems
- it is easily scalable