A Family of Fast Syndrome Based Cryptographic Hash Functions

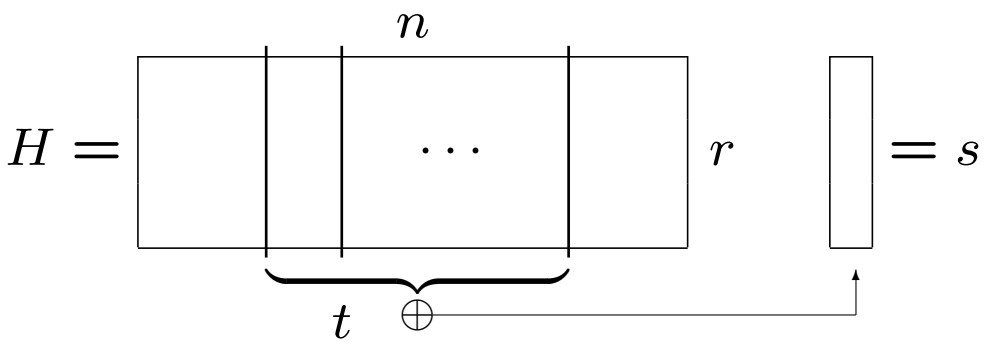
Nicolas Sendrier
(joint work with Daniel Augot and Matthieu Finiasz)

INRIA Rocquencourt, projet CODES

Ecrypt Conference – Hash Functions

Syndrome computing – A fast primitive

Let H be a binary $r \times n$ matrix



$$W_{n,t} = \{e \in \{0,1\}^n \mid w_H(e) = t\}$$

$$S : \begin{array}{ccc} W_{n,t} & \longrightarrow & \{0,1\}^r \\ e & \longmapsto & eH^T \end{array}$$

Problem: Syndrome Decoding NP-complete

Instance: An $r \times n$ binary matrix H, a word s of $\mathbf{F}_2^r$ and an integer $t > 0$.

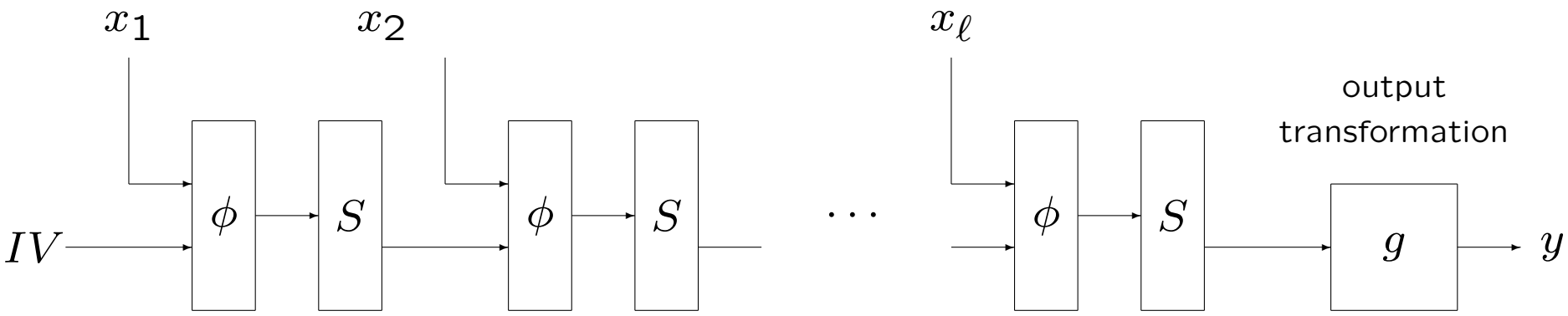
Question: Is there a word of weight $\leq t$ in $\{e \mid eH^T = s\}$?

Compression fonction

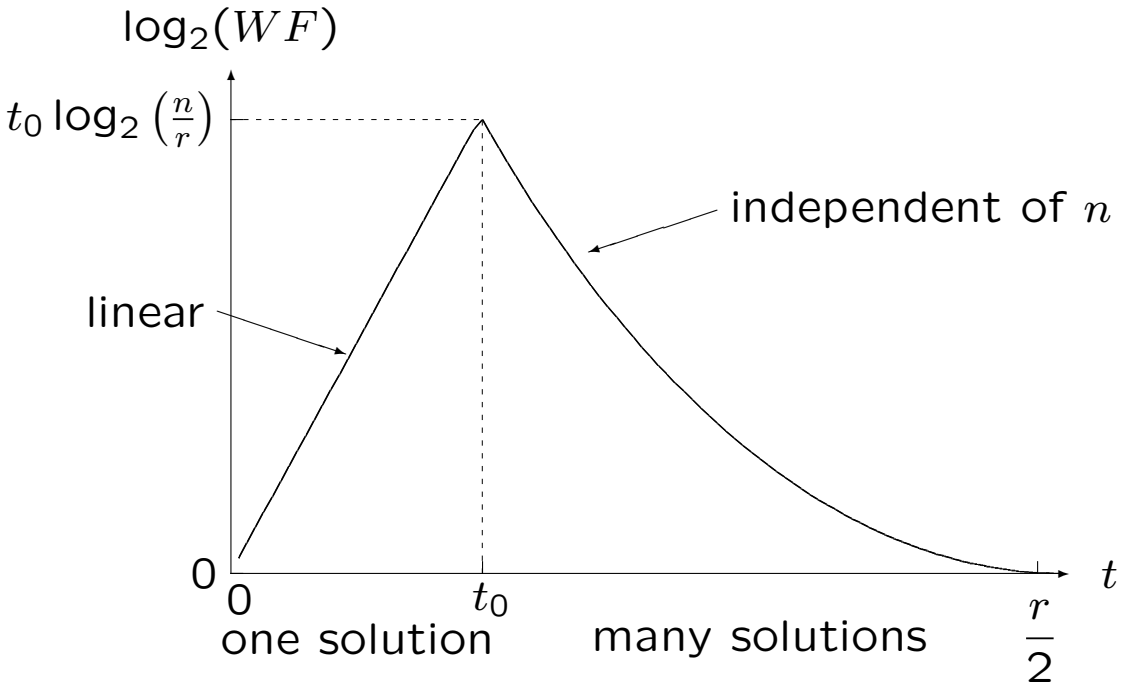
To achieve compression, we must have $\binom{n}{t} > 2^r$. We must also define an encoding procedure (possibily with variable length input) to transform binary data in words of constant weight and length

$$\phi : \{0,1\}^* \longrightarrow W_{n,t}$$

With the classical iterated construction, the text $(x_1, \ldots, x_\ell)$ is hashed as follows (matrix H is chosen randomly once and for all):



Preimage resistance – Information set decoding



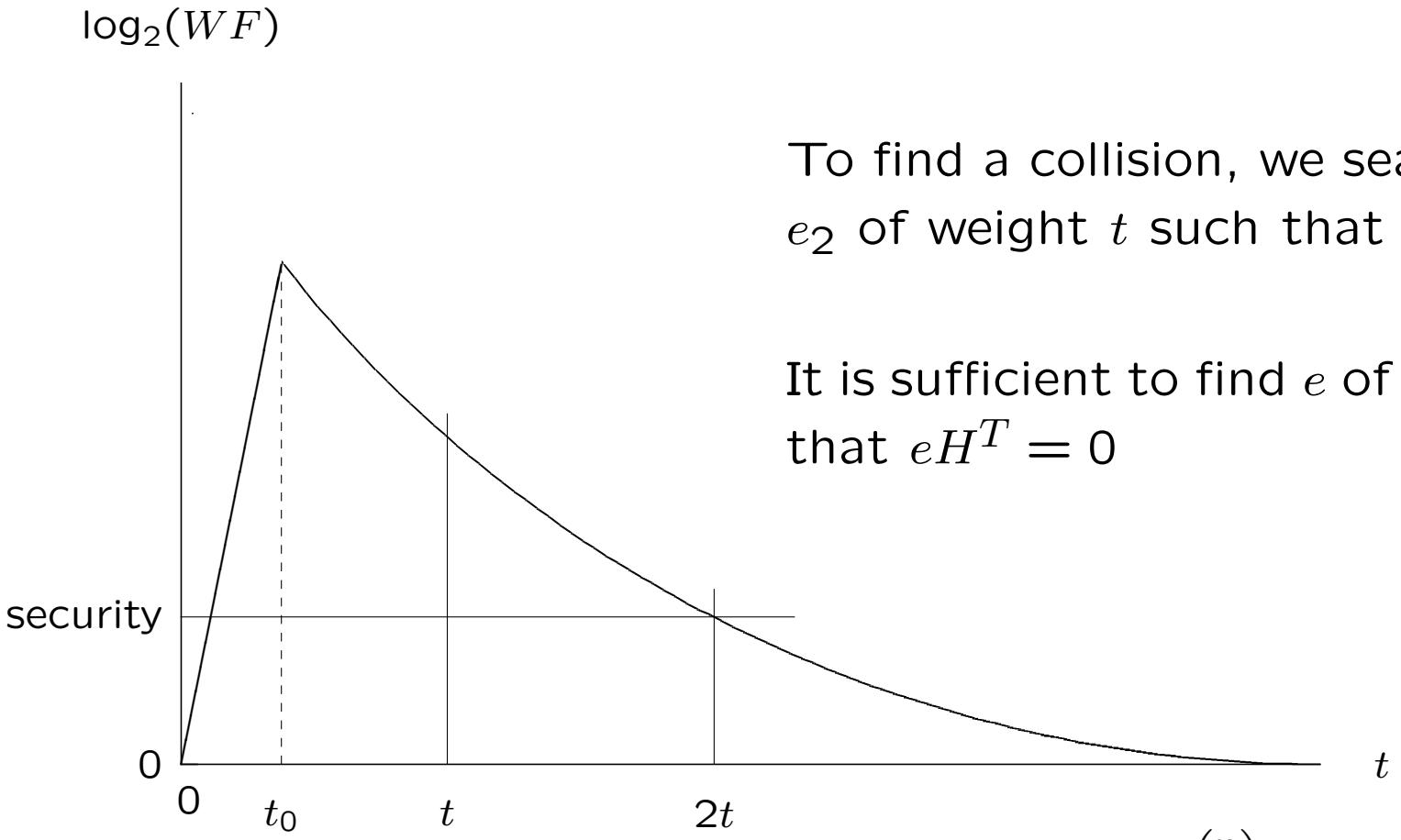
Cost for solving $s = eH^T$ for a given H and s, with e of weigth t by information set decoding.

Both n and r are fixed

t_0 is such that $\binom{n}{t_0} \approx 2^r$.

- The left hand side produces injective mappings (encryption)
- The point t_0 produces a bijective mapping (signature)
- The right hand side produces surjective mappings (compression)

Collision resistance



To find a collision, we search for e_1 and e_2 of weight t such that $e_1 H^T = e_2 H^T$.

It is sufficient to find e of weight $2t$ such that $eH^T = 0$

Trade-off between the compression ratio $\dfrac{\log_2 \binom{n}{t}}{r}$ and the security

Camion/Patarin/Wagner generalized birthday attack

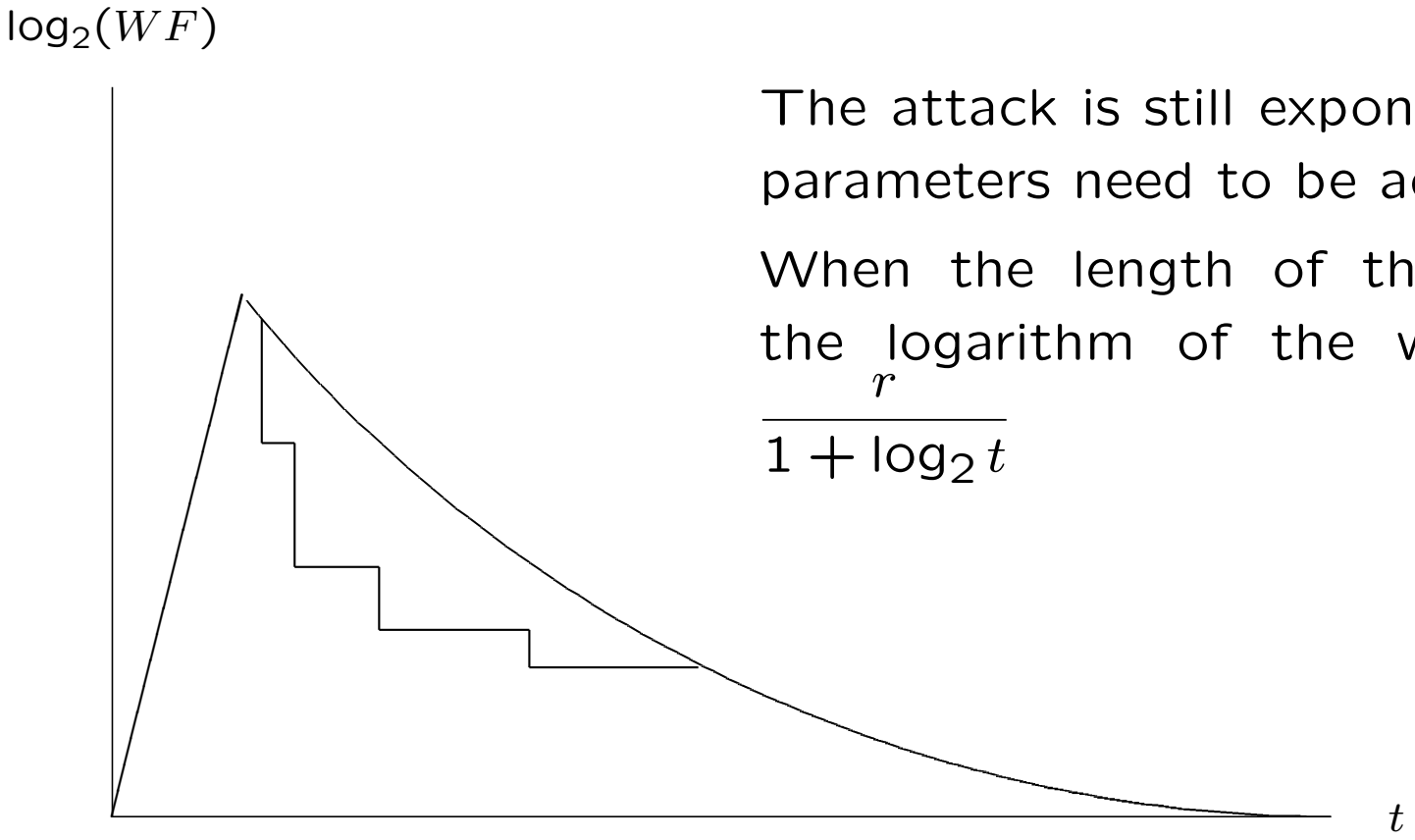
Unfortunately, information set decoding is not the best known attack (Coron and Joux).

The cost for solving syndrome decoding for weight w is $O(2^{\frac{r}{1+s}})$ where s is the largest integer such that

$$\begin{cases} s \leq \log_2 w \\ \binom{n}{w/2^s} > 2^{\frac{r}{1+s}} \end{cases}$$

When there are many solutions to the problem ($\binom{n}{2t} > 2^r$) then s can be larger than 1.

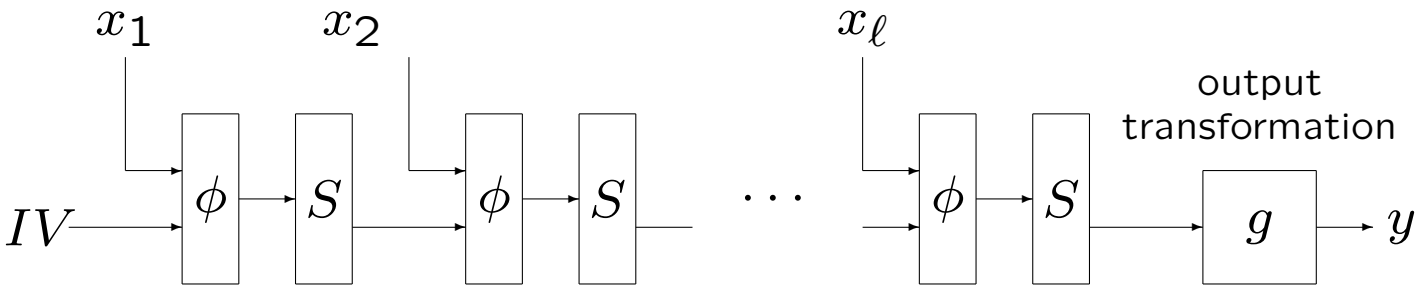
Generalized birthday attack



The attack is still exponential, but the parameters need to be adjusted.

When the length of the code grows the logarithm of the work factor is $\frac{r}{1+\log_2 t}$

Implementation – The encoding problem



$\phi : \{0,1\}^* \to W_{n,t}$ $\qquad$ $S : W_{n,t} \to \{0,1\}^r$

The mapping S is very fast and easy to implement, the encoding ϕ is more problematic

- slow: counting techniques
- fast: source coding techniques
- very fast: regular words

Encoding of constant weight words

- exact encoding: bijection between $W_{n,t}$ and $[0, \binom{n}{t}[$

$$(0 \leq i_1 < i_2 < \cdots < i_t < n) \Leftrightarrow \left(0 \leq \binom{i_1}{1} + \binom{i_2}{2} + \cdots + \binom{i_t}{t} < \binom{n}{t}\right)$$

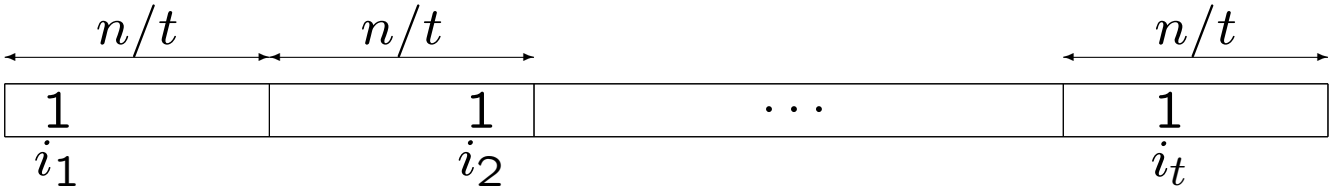
- source coding: variable length encoding of the strings of zeroes

  $\longrightarrow$ efficiency close to 1

- regular words: every bloc of n/t bits has weight exactly 1

  $\longrightarrow$ fast and easy to implement (particularly if n/t is a power of 2)

  $\longrightarrow$ efficiency is poor (i.e. fewer message bits processed each round)



$0 \leq i_j < n/t$

Security with regular words

- The general problem is still NP-complete
- Information set decoding is harder
- Generalized birthday attack is not harder

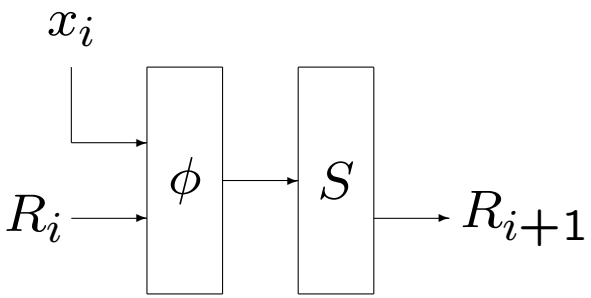
Example of parameters:

r	n/t	t	$\|H\|$	input of ϕ	bits/round	log security
320	256	42	3 Mbits	336 bits	16	80
400	256	85	8 Mbits	680 bits	280	80
480	256	170	20 Mbits	1360 bits	880	80

Implementation with regular words

Let us take $r = 400$, $t = 85$, $n = 256 \cdot t = 21760$

The matrix H is 400×21760. The input of ϕ is $t = 85$ bytes



$x_i = (a_0, \ldots, a_{34})$, 35 bytes of the message

$R_i = (a_{35}, \ldots, a_{84})$, 50 bytes or 400 bits

output of the previous round

The output R_{i+1} is obtained by adding bitwise the columns of H indexed by $a_j + 256 \cdot j$, $0 \leq j < 85$

(without regular words, we could input 49 *bytes of message each round instead of* 35*)*

Getting rid of the matrix

Increasing the throughput of the algorithm is possible by increasing the size of H, if we do that storing and accessing H becomes more and more problematic.

Since H is a random matrix, we can choose it pseudo-randomly and generate its columns "on demand".

Instead of providing H, we will provide a mapping which generates its columns ($f(i)$ will be the i-th column)

$$f : \{0, \ldots, n-1\} \to \{0,1\}^r$$

Let $e \in W_{n,t}$ with '1's in positions $0 \leq i_1 < i_2 < \cdots < i_t < n$.

We have $S(e) = eH^T = f(i_1) + f(i_2) + \cdots + f(i_t)$.

Virtual parity check matrix

Let $n = 2^m$, we have $f : \{0,1\}^m \to \{0,1\}^r$.

In addition to encoding and syndrome computing, everytime we iterate the compression fonction, we will make t calls to $f()$.

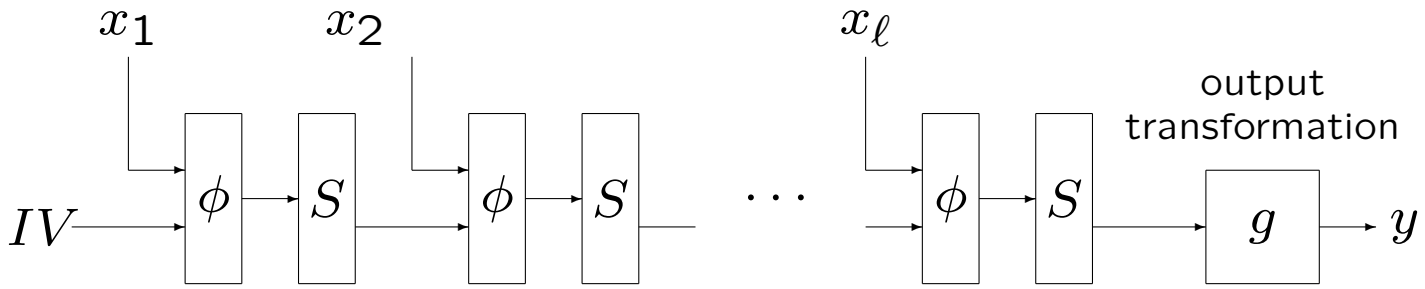
Those calls are likely to be dominant in the algorithmic cost, but, on the other hand, we have much more freedom on the parameters.

With regular words, $t(m - \log_2 t) - r$ message bits are processed at each iteration. For each call to $f()$, we process $m - \log_2 t - r/t$ bits. If we let the parameters grow, the hash function will be almost as efficient as $f()$ alone.

Virtual parity check matrix (instance)

We take $t = 8$, $r = m = 512$ and $f()$ a one-way permutation

With regular words, the encoder ϕ reads 4072 bits, and the syndrome function S outputs 512 bits. At each iteration we thus process 3560 bits of message



Security: the generalized birthday attack of order $s = \log_2(2t) = 4$ applies. The security exponent is $r/(1+s) \approx 102$.

Virtual parity check matrix (conclusions)

The security of this construction needs to be discussed further. Assume we use $f : \{0,1\}^m \rightarrow \{0,1\}^r$.

The question is: "What is the weakest assumption we have to make on $f()$ in order to obtain a collision-free hash function?"

Two observations:

- One must not be able to find $w \leq 2t$ elements $x_1, \ldots, x_w$ in $\{0,1\}^m$ such that $f(x_1) + f(x_2) + \cdots + f(x_w) = 0$
- If $f()$ is (as strong as) a block cipher, we already have good constructions

References

[AFS03] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A fast provably secure cryptographic hash function. Cryptology ePrint Archive, Report 2003/230, 2003. `http://eprint.iacr.org/`.

[CJ04] Jean-Sebastien Coron and Antoine Joux. Cryptanalysis of a provably secure cryptographic hash function. Cryptology ePrint Archive, Report 2004/013, 2004. `http://eprint.iacr.org/`.

[AFS05] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In *MyCrypt 2005*, October 2005.

[Sen05] Nicolas Sendrier. Encoding information into constant weight words. In *IEEE Conference, ISIT'05*, Adelaide, Australia, September 2005.