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

One-Way Functions

A one-way function is a function which is:

- simple to evaluate
  → should be as fast as possible
- hard to invert
  → ideally, with good security arguments

One-Way Functions

A one-way function is a function which is:

- simple to evaluate
  → should be as fast as possible
- hard to invert
  → ideally, with good security arguments

There are many applications of one-way functions in cryptography:

- compression functions to build cryptographic hash functions
- expansion functions for PRNG

One-Way Functions

A one-way function is a function which is:

- simple to evaluate
  → should be as fast as possible
- hard to invert
  → ideally, with good security arguments

There are many applications of one-way functions in cryptography:

- compression functions to build cryptographic hash functions
- expansion functions for PRNG

Unfortunately, one-way functions are hard to build:

- some are very fast, with few security arguments
- some have strong security arguments, but are slow

Niederreiter Encryption as a One-Way Function

Any public key encryption scheme is a one-way function:

- with a trapdoor (the decryption key)
- with strong security arguments

  → but public key encryption is usually slow

Niederreiter Encryption as a One-Way Function

Any public key encryption scheme is a one-way function:

- with a trapdoor (the decryption key)
- with strong security arguments
  → but public key encryption is usually slow

Niederreiter encryption is much faster than other public key schemes:

- convert the input to a low weight word
  → many different techniques for this
- compute its syndrome
  → only a few XORs, especially if the weight is very low

- The trapdoor can easily be removed
  → simply use a random binary matrix
- With a few tweaks it can be made even faster

Overview of the One-Way Function

Parameters:

- A binary $r \times n$ matrix H
- A constant weight encoding function φ from $\mathbb{F}_2^\ell$ to words of weight w in $\mathbb{F}_2^n$

Overview of the One-Way Function

Parameters:

- A binary $r \times n$ matrix H
- A constant weight encoding function φ from $\mathbb{F}_2^\ell$ to words of weight w in $\mathbb{F}_2^n$

One-Way Function:

- Take an input $x \in \mathbb{F}_2^\ell$
- Compute the output $y \in \mathbb{F}_2^r$ such that $y = H \times \varphi(x)$

Overview of the One-Way Function

Parameters:

- A binary $r \times n$ matrix H
- A constant weight encoding function φ from $\mathbb{F}_2^\ell$ to words of weight w in $\mathbb{F}_2^n$

One-Way Function:

- Take an input $x \in \mathbb{F}_2^\ell$
- Compute the output $y \in \mathbb{F}_2^r$ such that $y = H \times \varphi(x)$

Security

Inverting the function requires to solve an instance of Syndrome Decoding.

Efficiency

With φ fast and w small, the function can be very fast.

Fast Constant Weight Encoding

Exact encoding:

- maps an integer in $[1, \binom{n}{w}]$ to a word of weight w in F_2^n.
- requires computations on large integers

  $\rightarrow$ rather inefficient, but offers the largest possible input

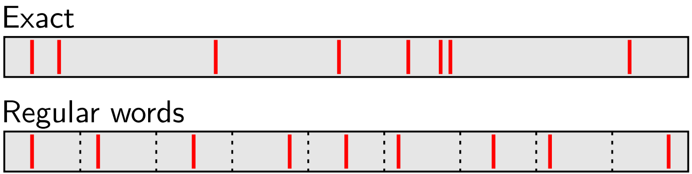
Fast Constant Weight Encoding

Exact encoding:

- maps an integer in $[1, \binom{n}{w}]$ to a word of weight w in $\mathbb{F}_2^n$.
- requires computations on large integers
  → rather inefficient, but offers the largest possible input

Regular words encoding:

- restrict to words with weight 1 in each interval of size $\frac{n}{w}$
- extremely fast if $\frac{n}{w}$ is a power of 2
  → the input space is smaller



A Fast One-Way Function

Input: x of $w \times \log \frac{n}{w}$ bits.

Algorithm:

- split x into w blocks of $\log \frac{n}{w}$ bits, convert each of them to integers $x_1, \ldots, x_w$
- for $i \in [1, w]$, pick column H_i at position x_i in the i-th interval of H
- return $y = H_1 \oplus \cdots \oplus H_w$

A Fast One-Way Function

Input: x of $w \times \log \frac{n}{w}$ bits.

Algorithm:

- split x into w blocks of $\log \frac{n}{w}$ bits, convert each of them to integers $x_1, \ldots, x_w$
- for $i \in [1, w]$, pick column H_i at position x_i in the i-th interval of H
- return $y = H_1 \oplus \cdots \oplus H_w$

Efficiency:

- in theory, splitting x has no cost
  → in practice, in software, depending on $\log \frac{n}{w}$, it can cost a few shifts/XORs per x_i
- the XORing costs $r \times w$ binary XORs
  → pick secure parameters with r and w small

Security of the Construction

Attacking the one-wayness of the function requires to solve a Syndrome Decoding instance.

Security of the Construction

Attacking the one-wayness of the function requires to solve a Syndrome Decoding instance.

⚠ But not any instance: a regular instance

Security of the Construction

Attacking the one-wayness of the function requires to solve a Syndrome Decoding instance.

⚠ But not any instance: a regular instance

Two possible approaches to measure the security of such instances:

- tweak ISD/GBA attacks for regular instances
  → hard to know if the absolute best attack was found
- loosely bound the security
  → the security drop can't be more than the probability of a word to be regular

Security of the Construction

Attacking the one-wayness of the function requires to solve a Syndrome Decoding instance.

⚠ But not any instance: a regular instance

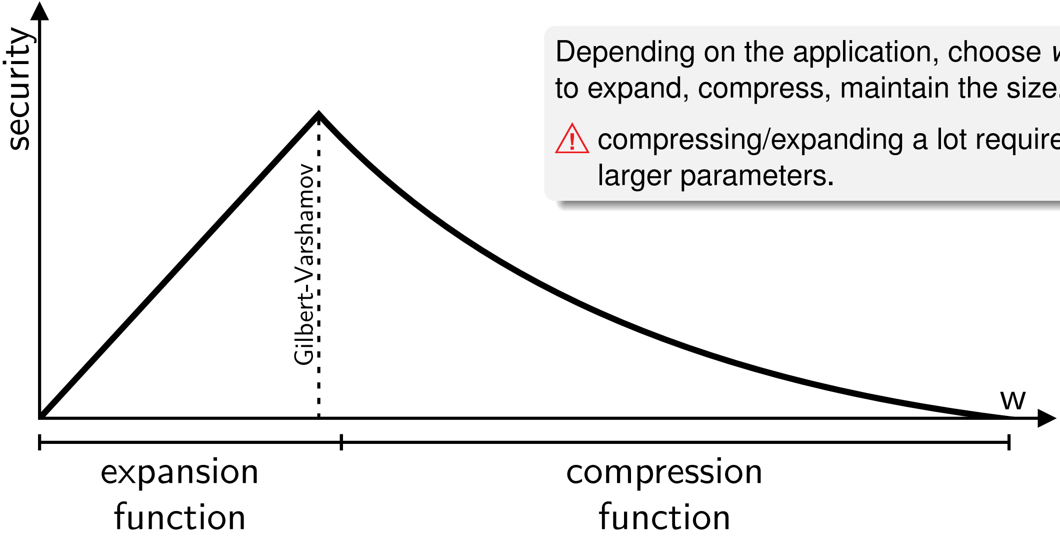
Two possible approaches to measure the security of such instances:

- tweak ISD/GBA attacks for regular instances
  → hard to know if the absolute best attack was found
- loosely bound the security
  → the security drop can't be more than the probability of a word to be regular

Regular Syndrome Decoding Security

$$\text{Security(regular SD)} \geq \text{Security(SD)} \times \underbrace{\frac{\left(\frac{n}{w}\right)^w}{\binom{n}{w}}}_{\simeq \frac{w!}{w^w}}$$

Parameter Selection



Depending on the application, choose *w* to expand, compress, maintain the size.

⚠ compressing/expanding a lot requires larger parameters.

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**