The Niederreiter Cryptosystem

This cryptosystem was introduced by H. Niederreiter in 1985. The security relies directly upon SDP and hence it is often considered a "dual" version of the original McEliece cryptosystem.

Table 1: The Niederreiter cryptosystem

Setup	Fix public system parameters $q, m, n, k, w \in \mathbb{N}$ such that $k \geq n - wm$.
K	$\mathsf{K}_{\mathsf{publ}}$ the set of $k \times n$ matrices over $\mathbb{F}_q$. $\mathsf{K}_{\mathsf{priv}}$ the set of triples formed by a $k \times k$ invertible matrix, an $n \times n$ permutation matrix and a code description.
P	The set $\mathbb{W}_{n,w}$ of words of $\mathbb{F}_q^n$ with Hamming weight w.
C	The vector space $\mathbb{F}_q^{(n-k)}$.
KeyGen	Generate at random a polynomial $g \in \mathbb{F}_{q^m}[x]$ and elements $\alpha_1, \dots, \alpha_n \in \mathbb{F}_{q^m}$, then build the Goppa code $\varGamma = \varGamma(\alpha_1, \dots, \alpha_n, g)$ over $\mathbb{F}_q$ and its parity-check matrix H. Select at random a $k \times k$ invertible matrix S and an $n \times n$ permutation matrix P. Publish the public key $\hat{H} = SHP \in \mathsf{K}_{\mathsf{publ}}$ and store the private key $(S, P, \varGamma) \in \mathsf{K}_{\mathsf{priv}}$.
Enc	On input a public key $\hat{H} \in \mathsf{K}_{\mathsf{publ}}$ and a plaintext $\underline{e} \in \mathsf{P}$, compute the syndrome of $\underline{e}$, that is $\underline{s} = \hat{H}\underline{e}^T$ and return the ciphertext $\psi = \underline{s} \in \mathsf{C}$.
Dec	On input the private key $(S, P, \varGamma) \in \mathsf{K}_{\mathsf{priv}}$ and a ciphertext $\psi \in \mathsf{C}$, first compute $S^{-1}\psi$ then apply the decoding algorithm $\mathsf{D}_\varGamma$ to it. If the decoding succeeds, multiply the output $\underline{\hat{e}}$ by P^{-1}, and return the resulting plaintext $\phi = P^{-1}\underline{\hat{e}}$. Otherwise, output $\perp$.

Just like before, it is easy to verify the consistency of the decryption process. In fact, $S^{-1}\psi = HP\underline{e}$ and since P is a permutation matrix, the vector $P\underline{e}$ has still weight w. Decoding and then multiplying by P^{-1} on the left returns the desired plaintext.

The computational assumptions for Niederreiter are almost the same, except for the indistinguishability assumption, that changes as follows.

Assumption 1 (Indistinguishability) *The $(n-k) \times n$ matrix $\hat{H}$ output by* KeyGen *is computationally indistinguishable from a uniformly chosen matrix of the same size.*

Remark 1 Note that the use of matrices S and P, in both schemes, is rather outdated and unpractical; moreover, it can introduce vulnerabilities to the scheme as per the work of Strenzke et al.. A still secure, but much simpler description would be to take the public key $\hat{G}$ (resp. $\hat{H}$) to be just the systematic form of G (resp. H), and the private key to be $\varGamma$ alone.

The encryption process for both cryptosystems is very fast. In fact, its complexity is dominated by (McEliece) or exactly equal to (Niederreiter) a matrix-vector multiplication operation. Even intuitively, this is much simpler than, for example, exponentiation such as the case of RSA.

Recent benchmarks suggest that the McEliece encryption process is often even faster than the NTRU cryptosystem, which makes of fast encryption its strongest point. Decryption, on the other hand, involves a decoding operation and that increases considerably the complexity time.
The major drawback of the scheme is the large memory requirements, in particular the necessity to store a big public key. This is possibly also the main reason why code-based cryptography has not yet been considered in any practical application. McEliece in the original manuscript sets the parameters as $n = 1024, k = 524, w = 50$. With this setting, the public key size is 524×1024 bits $= 67,072$ bytes.

A first improvement comes already with the Niederreiter scheme, following the suggestion to compute the systematic form of the public key, i.e. $\hat{H} = (M|I_{n-k})$, and store only the non-trivial part M to save some space. This would require 500×524 bits $= 32,750$ bytes, clearly still too big for most applications.

Several proposals have then been made in the following years, trying to modify McEliece's original framework in order to deal with this issue. Unfortunately, almost all of them turned out to be insecure or inefficient. Niederreiter himself, in the first place, suggests to use GRS codes instead of Goppa codes for his scheme. A famous attack due to Sidelnikov and Shestakov was subsequently published in 1992 and proved that the algebraic structure of GRS codes can be easily exploited, de facto excluding the whole class from the possible choices for a coding-theory based scheme. A similar fate occurred to proposals centered on Reed-Muller codes and Gabidulin codes.

It is necessary to choose carefully the family of codes used to generate the keys. For the original McEliece cryptosystem, for example, Goppa codes with a binary generator polynomial produce weak keys. Loidreau and Sendrie show that these instances are easily recognizable: in fact, the automorphism group of a Goppa code with binary generator polynomial is generated by the Frobenius field automorphism. This results in an attack that makes use of Sendrier's Support Splitting Algorithm (SSA). Other unsuccessful attempts include, for example, concatenated codes, elliptic codes and the algebraic-geometric codes proposed by Janwa and Moreno, although for the latter only the case of curves with small genus has been cryptanalysed properly.

The general pitfalls to avoid are twofold:

- Families with high performance, like the above cited concatenated codes, turbo-codes or LDPC codes, are likely to leak some structure due to the high number of low-weight codewords in their duals.

- Families having optimal (as for the GRS codes) or sub-optimal (elliptic codes) combinatorial properties like Maximum Distance Separable (MDS) are also dangerous, since minimum-weight codewords are not hard to find and reveal a lot of information about the code structure.

Recent Proposals

Since codes with too much evident algebraic structure don't seem to provide a secure choice for McEliece, a new approach is instead being attempted. It consists of introducing just a partial algebraic structure, using clever scrambling techniques to preserve it, while hoping to hide enough of the underlying private code. The core idea is to make use of subfield subcodes.

Quasi-Cyclic

A first example in this direction was given in 2005 by Gaborit and further pursued by Berger, Cayrel, Gaborit and Otmani. The scheme makes use of the so-called quasi-cyclic codes.

Definition 1 Let $N = N_0\ell$ and let π_ℓ be the permutation on $\{0, \dots, N-1\}$ defined by the orbits $\{(0, \dots, \ell-1), (\ell, \dots, 2\ell-1), \dots, ((N_0-1)\ell, \dots, N-1)\}$. A linear code $\mathcal{C}$ of length N is *Quasi-Cyclic* of order ℓ and index N_0 if it is globally invariant under the action of π_ℓ.

Similarly to cyclic codes, this family can be described by means of a matrix in circulant form, although in this case the matrix will be only block-circulant.

The key generation process starts by choosing a Reed-Solomon code in quasi-cyclic form defined over a large alphabet $\mathbb{F}_{q^m}$. This is easy since it's well known that every Reed-Solomon code is in fact a cyclic code; all one needs to do then is to rearrange the support in order to get a quasi-cyclic code. After rearranging and deleting the majority of the blocks (to counter key-recovery attacks tied to the quasi-cyclic structure), the next step consists of transforming the shortened Reed-Solomon code into a quasi-cyclic Generalised Reed-Solomon code. This is accomplished purely by algebraic means by scalar multiplication and matrix multiplication with a diagonal matrix. Finally, the subfield subcode is constructed over $\mathbb{F}_q$ and the resulting block-circulant matrix is the public key.

Quasi-Dyadic

This scheme was presented by Misoczki and Barreto in 2009 and it features a structure similar to the quasi-cyclic proposal, but using codes in quasi-dyadic form instead.

Definition 2 Given a ring R and a vector $\underline{h} = (h_0, \dots, h_{n-1}) \in \mathsf{R}^n$, the *Dyadic* matrix $\Delta(\underline{h}) \in \mathsf{R}^{n\times n}$ is the symmetric matrix with components $\Delta_{ij} = h_{i\oplus j}$, where $\oplus$ stands for bitwise exclusive-or on the binary representations of the indices. The sequence $\underline{h}$ is called its signature. Moreover, $\Delta(t, \underline{h})$ denotes the matrix $\Delta(\underline{h})$ truncated to its first t rows. Finally, a matrix is *Quasi-Dyadic* if it is a block matrix whose component blocks are $t \times t$ dyadic submatrices.

If n is a power of 2, then every $2^k \times 2^k$ dyadic matrix can be described recursively as

$$\Delta = \begin{pmatrix} A & B \\ B & A \end{pmatrix} \tag{1}$$

where each block is a $2^{k-1} \times 2^{k-1}$ dyadic matrix (and where any 1×1 matrix is dyadic).

Definition 3 Let Π_i be the dyadic matrix $\Delta(\underline{h})$ whose signature $\underline{h}$ is the i-th row of the identity matrix. This is called *dyadic permutation* since it is a permutation matrix that preserves the dyadic structure.

The scheme is based on Goppa codes as in the original McEliece, but these are carefully selected to admit a parity-check matrix in Cauchy form.

Definition 4 Given two disjoint sequences $\underline{v} = (v_1, \dots, v_\ell) \in \mathbb{F}_{q^m}^\ell$ and $\underline{L} = (L_1, \dots, L_n) \in \mathbb{F}_{q^m}^n$, the *Cauchy matrix* $C(\underline{v}, \underline{L})$ is the matrix with components $C_{ij} = \dfrac{1}{v_i - L_j}$, i.e.

$$C(\underline{v}, \underline{L}) = \begin{pmatrix} \dfrac{1}{v_1 - L_1} & \dots & \dfrac{1}{v_1 - L_n} \\ \vdots & \vdots & \vdots \\ \\ \dfrac{1}{v_\ell - L_1} & \dots & \dfrac{1}{v_\ell - L_n} \end{pmatrix}. \tag{2}$$

Cauchy matrices have the property that all of their submatrices are invertible. Note that in general Cauchy matrices are not necessarily dyadic and vice-versa, but the intersection of these classes is non-empty in characteristic 2.

Goppa codes admit a parity-check matrix in Cauchy form if the generator polynomial is monic and without multiple zeros. In particular, the following theorem holds.

Theorem 1 *Let $\Gamma = \Gamma(\alpha_1, \dots, \alpha_n, g)$ be a Goppa code. If the generator polynomial g is monic and separable, i.e. $g(x) = (x - x_0) \dots (x - x_{\ell-1})$, then Γ admits a parity-check matrix in Cauchy form $H = C(\underline{x}, \underline{\alpha})$.*

The trick to generate a public key in dyadic form is to choose a Goppa code that allows a parity-check matrix that is simultaneously dyadic and Cauchy. Misoczki and Barreto show that this intersection is non-empty.
The core idea is to start from a fully dyadic code, and then select, permute and scale the columns (block by block) before applying the subfield subcode technique.
All the operations involved preserve the dyadicity of the matrix, including the use of dyadic permutations, the co-trace function, and the block operations performed during the final Gaussian elimination. In this way, the public key will be composed of dyadic submatrices each of which can be represented compactly by its signature, therefore saving a factor of ℓ in the public key size. Since M is $(n - k) \times k = m\ell \times k$ and is $\ell \times \ell$ block dyadic, it requires only $km\ell/\ell = km$ field elements for storage, equivalent to kmc bits.

FOPT

Both proposals achieve very good parameters and key lengths, but unfortunately are susceptible to the following structural attack proposed by Faugère, Otmani, Perret and Tillich. It relies on the fundamental property of coding theory $H \cdot G^T = 0$ to build an algebraic system, using then Gröbner bases techniques to solve it. The attack in its purest form is applicable also to the original McEliece, although in general it wouldn't be possible to perform it efficiently. The special properties of the structured codes used in the variants presented above are of key importance, as they contribute to considerably reduce the number of unknowns of the system. The algorithm takes into account that the codes in use are part of the alternant family, and therefore it looks for a parity-check matrix that, even if different from the private key, still allows efficient decoding.

Table 2: The FOPT algorithm

Input	A $k \times n$ generator matrix $G = \{g_{i,j}\}$ for the subcode $\mathcal{C}\|_{\mathbb{F}_q}$, G being a matrix formed of $\ell \times \ell$ blocks, with $k = k_0\ell, n = n_0\ell$, over $\mathbb{F}_q = \mathbb{F}_{2^c}$.
Output	A parity-check matrix in alternant form $H = \{y_i x_i^j\}$ for $\mathcal{C}$ over $\mathbb{F}_{q^m}$.
1.	Generate the following system of equations in the unknowns $X = \{X_i\}$ and $Y = \{Y_i\}$: $$\left\{g_{i,0}Y_0X_0^j + \cdots + g_{i,n-1}Y_{n-1}X_{n-1}^j = 0 \mid 0 \le i \le k-1, 0 \le j \le \ell-1\right\}. \quad (3)$$
2.	Choose $n_{Y'} \ge n-k$ variables Y' from Y, and use the equations to express all other variables in $Y \backslash Y'$ as polynomials in Y'. The Y' variables are called "free" and the remaining "dependent".
3.	Compute the projection of the solutions with respect to the variables Y'.
4.	Having determined the Y', the system will simplify to $$\left\{g'_{i,0}X_0^j + \cdots + g'_{i,n-1}X_{n-1}^j = 0 \mid 0 \le i \le k-1, 0 \le j \le \ell-1\right\}. \quad (4)$$
5.	Consider now the subset of the equations having degree equal to a power of two, i.e. $j = 2^l$, for $l = 1, \ldots, \log_2(\ell-1)$.
6.	Use the Frobenius automorphism to produce a system over $\mathbb{F}_2$, consisting of mcn unknowns and $mc\log_2(\ell-1)k$ equations.
7.	Solve the system for X_i and return H.

Observe that for all suitable choices of cryptographic parameters, we have that $\log_2(\ell-1)k > n$, hence the system produced in Step 6 is easily solvable.
Some relations are derived from the special properties peculiar of each of the two schemes, and then used in the context of Step 2 to determine the number of free variables $n_{Y'}$ and simplify the system. These are presented below:

Table 3: Properties for the quasi-cyclic (left) and quasi-dyadic (right) schemes.

$\begin{cases} X_{j\ell+i} = X_{j\ell}\beta^i \\ Y_{j\ell+i} = Y_{j\ell}\beta^{ie} \end{cases}$	$\begin{cases} Y_{j\ell+i} = Y_{j\ell} \\ X_{j\ell+i} + X_{j\ell} = X_i + X_0 \\ X_{j\ell+(i\oplus i')} = X_{j\ell+i} + X_{j\ell+i'} + X_{j\ell} \end{cases}$
for $0 \le j \le n_0 - 1$, $0 \le i \le \ell - 1$ and an integer $e \in \{0, \ldots, \ell-1\}$ picked secretly[1].	for $0 \le j \le n_0 - 1$ and $0 \le i, i' \le \ell - 1$.

Now, in some cases this number is very small (e.g. 1 or 2); an exhaustive search thus leads already to a practical attack. Otherwise, the technique used is to find a projection of the solutions with respect to the variables of the block Y', which can be done, as anticipated, by computing a Gröbner basis. This is by far the most expensive part of the algorithm.
Applying the relations to the general framework it is possible to deduce the following scenarios for the two schemes, where r such that $rm = n - k$ is the order of the alternant form (in the quasi-dyadic case $r = \ell$):

[1]For the purpose of the attack, e is assumed to be known, even just via an exhaustive search.

Table 4: System specifications for the quasi-cyclic (left) and quasi-dyadic (right) schemes.

	QC	QD
Unknowns Y_i	$n_0 - 1$	$n_0 - 1$
Unknowns X_i	$n_0 - 1$	$n_0 - 2 + \log_2 \ell$
Linear equations involving only the Y_i	k_0	$n_0 - m$
Nonlinear equations containing monomials of the form $Y_i X_i^\eta$, $\eta > 0$	$(r-1)k_0$	$\ell(\ell-1)(n_0-m)$

Further analysis has been conducted by the same authors in a successive work, and a theoretical estimate has been provided. The experimental results obtained by running the algorithm on the set of parameters proposed in the previous schemes prove to be reasonably close to this bound, even if this is far from being tight. At the current time, no further analysis has been conducted and the numbers provided by the bound can be interpreted as a good approximation of the overall costs of the algorithm. While this clearly doesn't fully assess the security of the scheme, it is enough to discard many weak sets of parameters.

Table 5: Summary of the complexities for the FOPT attack.

	Name	q	m	ℓ	n_0	$n_{X'}$	$n_{Y'}$	Security	Time(s)	Ops	T_{theo}
QC	A_{16}	2^8	2	51	9	8	3	80	0.06	$2^{18.9}$	2^{17}
	B_{16}				10	9	3	90	0.03	$2^{17.1}$	2^{18}
	C_{16}				12	11	3	100	0.05	$2^{16.2}$	2^{20}
	D_{16}				15	14	4	120	0.02	$2^{14.7}$	2^{26}
	A_{20}	2^{10}	2	75	6	5	2	80	0.05	$2^{15.8}$	2^{10}
	B_{20}			93	6	5	2	90	0.05	$2^{17.1}$	2^{10}
	C_{20}			93	8	7	2	110	0.02	$2^{14.5}$	2^{11}
	QC_{600}	2^8	2	255	15	14	3	600	0.08	$2^{16.6}$	2^{21}
QD	Table 2	2^2	8	64	56	59	7	128	1776.3	$2^{34.2}$	2^{65}
		2^4	4		32	36	3		0.50	$2^{22.1}$	2^{29}
		2^8	2		12	16	1		0.03	$2^{16.7}$	2^8
	Table 3	2^8	2	64	10	14	1	102	0.03	$2^{15.9}$	2^8
				128	6	11		136	0.02	$2^{15.4}$	2^7
				256	4	10		168	0.11	$2^{19.2}$	2^7
	Table 5	2^8	2	128	4	9	1	80	0.06	$2^{17.7}$	2^6
				128	5	10		112	0.02	$2^{14.5}$	2^7
				128	6	11		128	0.01	$2^{16.6}$	2^7
				256	5	11		192	0.05	$2^{17.5}$	2^7
				256	6	12		256	0.06	$2^{17.8}$	2^7

Overall the attack is very successful in almost all cases. The complexity increases proportionally to the value $\rho = m - 1$ (at least for the quasi-dyadic case), so unlike the other case, it does not depend on the code parameters (length, dimension) but on the field chosen.
For some of the parameters $\rho = 15$ and the time necessary for the computation is beyond the range of the machine in use for the tests (Xeon bi-processor 3.2Ghz, with 16 Gb of Ram). The authors therefore conclude that any system with $\rho \leq 20$ should be within the scope of the attack.

Remark 2 For binary quasi-dyadic Goppa codes the analysis is less accurate. In this case, in fact, it is easy to compute the Gröbner basis for the system, but this is somehow "trivial", i.e. reduced to just one equation, therefore not providing enough information. This is typically due to the fact that only a subset of the equations is used (the ones with bi-degree $(2^j, 1)$). As a result, the variety associated is too big, and the attack cannot be mounted efficiently. To understand how to use all the equations in a more clever way in this case, remains an open problem.

Signatures

The idea of Courtois, Finiasz and Sendrier ins to use the hash-and-sign paradigm, sequentially adding an integer c to the input[2] of the hash function $\mathcal{H}$ and testing if $\mathcal{H}(\underline{\mu}, c)$ is a decodable syndrome, iterating the procedure until such a syndrome is found.
It is clear that in general the process is not efficient: with the original McEliece parameters $n = 1024, k = 524, w = 50, q = 2$ there are exactly 2^{500} distinct syndromes, of which only $\sum_{i=1}^{w} \binom{n}{i} \approx 2^{284}$ are decodable. Thus an average of 2^{216} decoding attempts would be needed, which is obviously not plausible. Even if the parameters are adjusted like suggested by the authors ($m = 16, w = 9$ with $n = 2^m, r = mw$, for a security level of 2^{80}), the scheme is far from practical: in order to sign it is necessary to repeat the algorithm in average 9! times, with the additional disadvantage of a very big public key (1152 Kbytes). On the other hand, the verification is intuitively very fast (similarly to the Niederreiter encryption process, it is just a matrix-vector multiplication) and the signature size can be considerably shortened, thanks to an indexing trick, to reach a size comparable to other schemes; however, the two above flaws are so limiting that the disadvantage of CFS is still too great in many applications.

A better picture can be seen from the following table, that includes three variants of the above CFS set of parameters ($m = 16, w = 9$). These are simple trade-off techniques optimized for fast verification (CFS1), short signature (CFS3) or halfway between the two (CFS2).

Table 6: Comparison of some of the most popular signature schemes. The signature size is given in bits, the public key size in kilobytes and all the timings are measured on a machine running at 1GHz. 'The column 'Data Size" specifies the instance of the scheme, for example the size of the RSA modulus in FDH, the syndrome length in CFS etc.

Hard Problem	Factoring	Disc. Log	Ell. Curves	Syndrome Decoding		
Scheme	RSA-FDH	DSA	ECDSA	CFS1	CFS2	CFS3
Data Size (bits)	1024	160/1024	160	144	144	144
Security	2^{80}	2^{80}	2^{80}	2^{80}	2^{80}	2^{80}
Signature (bits)	1024	320	321	132	119	81
Public Key (Kb)	0.2	0.1	0.1	1152	1152	1152
Signing	9 ms	1.5 ms	5 ms	10-30s	10-30s	10-30s
Verification	9 ms	2 ms	6 ms	< 1 μs	< 1 ms	≈ 1s

Due to a Generalized Birthday Attack (GBA) by Bleichenbacher, the original set of parameters ($m = 16, w = 9$) can't be considered secure anymore. The authors propose instead ($m = 21, w = 10$) or ($m = 15, w = 12$). Barreto, Cayrel, Misoczki and Niebuhr presented a variant of CFS using the quasi-dyadic framework. The public key size is considerably reduced and, among the various trade-offs between key size and signing complexity, intermediate choices seem the more appropriate: for example ($m = 15, w = 12$) results in a public key of 169 Kbytes, although in average $2^{29.8}$ repetitions are needed to sign, which is still very high.

[2]This can be realized, for example, by concatenating μ and the bit string corresponding to the integer c.

Identification Schemes

It would be great to be able to create an identification scheme following the simple framework introduced for lattices. One could think to translate the framework directly. This would be based on Syndrome Decoding, hence featuring a public matrix H, a secret $\underline{s}$ and the public key $\underline{S} = H\underline{s}^T$. The scheme would need to satisfy precise requirements:

- The secret $\underline{s}$ should have weight below the GV bound. This is to ensure that the secret is unique and that SD is hard.
- The final verification should include an algebraic formula consisting of H, the commitment $\underline{Y}$ and $\underline{S}$, plus a check on the weight of the response $\underline{z}$.
- The challenge c should be such that $c\underline{s}$ is defined, $\underline{z} = \underline{y} + c\underline{s}$ is defined and c is compatible with the final verification[3].

Note that the syndrome computation is preimage-resistant only if the weight of $\underline{x}$ is small. It follows that the scheme is subject to additional constraints. For example, the random element $\underline{y}$ and the challenge c should be chosen such that $\mathsf{wt}(\underline{z}) \leq w$, where w is the value of the GV distance. A natural choice for c is to be an element of $\mathbb{F}_q$. Since multiplication by a field element is an invariant for the Hamming weight, this means that $\underline{s}$ and $\underline{y}$ must satisfy $\mathsf{wt}(\underline{s}) = \gamma_1 w, \mathsf{wt}(\underline{y}) = \gamma_2 w$, for certain constants $\gamma_1, \gamma_2 \leq 1$ such that $\gamma_1 + \gamma_2 = 1$.
A sample instantiation is described below (for the case $\gamma_1 = \gamma_2 = 1/2$).

Table 7: Syndrome-based Identification Scheme.

Public Data	The parameters $q, n, k, w \in \mathbb{N}$ and an $(n-k) \times n$ parity-check matrix H over $\mathbb{F}_q$.
Private Key	$\underline{s} \in \mathbb{W}_{q,n,w/2}$.
Public Key	$\underline{S} = H\underline{s}^T$.

PROVER		VERIFIER
Choose $\underline{y} \overset{\$}{\leftarrow} \mathbb{W}_{q,n,w/2}$ and set $\underline{Y} = H\underline{y}^T$.	$\xrightarrow{\underline{Y}}$	
	$\xleftarrow{c}$	$c \overset{\$}{\leftarrow} \mathbb{F}_q \setminus \{0\}$.
Compute $\underline{z} = \underline{y} + c\underline{s}$.	$\xrightarrow{\underline{z}}$	Accept if $H\underline{z}^T = \underline{Y} + c\underline{S}$ and $\mathsf{wt}(\underline{z}) \leq w$.

The protocol is complete and well-defined. However, the conditions on the weight of $\underline{s}, \underline{y}$ and $\underline{z}$ make the scheme vulnerable to an attacker who tries to learn the secret.

Theorem 2 *Any identification scheme that satisfies the above requirements cannot be a zero-knowledge protocol.*

Proof It is possible to build an attacker $\mathcal{A}$ that will compute the private key. $\mathcal{A}$ is a passive attacker, that is, $\mathcal{A}$ has only evidence of the exchanges between $\mathcal{P}$ and $\mathcal{V}$: this is the weakest possible adversarial model. To recover the secret information $\underline{s}$, $\mathcal{A}$ runs the protocol several times. At every run of the protocol, it stores the challenge c and the corresponding response $\underline{z} = \underline{y} + c\underline{s}$,

[3]This means $Hc = c'H$ for some c' not necessarily equal to c.

then computes $\underline{z}' = c^{-1}\underline{y} + \underline{s}$. Note that this is always possible, since c is a field element and is non-zero. Without loss of generality, consider $\underline{z}'$ to be of the form $\underline{y}' + \underline{s}$. If $\underline{y}$ is randomly generated, so is $\underline{y}'$; moreover, $\mathsf{wt}(\underline{y}') = \mathsf{wt}(\underline{y}) = w/2 << n/2$ by construction, so each of the coordinates $\underline{y}'_i$ is biased to be more likely 0 than non-zero. Therefore, $\mathcal{A}$ can perform successfully a very simple statistical analysis: it fixes a particular i and observes the behavior of $\underline{y}'_i$ for multiple runs. If $\underline{y}'_i$ is non-zero most of the times, then $i \in \mathrm{supp}(\underline{s})$. Eventually, $\mathcal{A}$ is able to fully recover the support of $\underline{s}$. This completes the attack in the binary case, and gives enough information to recover $\underline{s}$ even in the non-binary case (for example with a general decoding algorithm). $\triangle$

The crucial point is that $\underline{y}$ is constrained to be of small weight, hence the expression $\underline{y} + c\underline{s}$ is not enough to properly hide the support of $\underline{s}$. This clashes with the other security requirement, that is, to avoid forgeries. If one drops the condition on $\mathsf{wt}(\underline{z})$, in fact, it is easy to find a vector that satisfies the verification equation. Thus it is infeasible to create a scheme in such a direct way.

The issue seems to come from the Hamming metric, which is too constraining to be able to hide the secret. Unlike the lattice case, in fact, vectors in the Hamming metric are measured on a position-dependent basis rather than on Euclidean norm. This seems to be leaking too much information to provide zero-knowledge.

Thus, so far, the only proposal still valid (along with its variants) is Stern's (1993).

Table 8: Stern Identification Scheme.

Public Data	The parameters $n, k, w \in \mathbb{N}$, an $(n-k) \times n$ parity-check matrix H over $\mathbb{F}_2$ and a hash function $\mathcal{H}$.
Private Key	$\underline{s} \in \mathbb{W}_{2,n,w}$.
Public Key	$\underline{S} = H\underline{s}^T$.

PROVER		VERIFIER
Choose $\underline{y} \xleftarrow{\$} \mathbb{F}_2^n$ and a permutation $\pi \xleftarrow{\$} \mathrm{Sym}(n)$, then set $c_1 = \mathcal{H}(\pi, H\underline{y}^T)$, $c_2 = \mathcal{H}(\pi(\underline{y})), c_3 = \mathcal{H}(\pi(\underline{y} + \underline{s}))$.	$\xrightarrow{c_1, c_2, c_3}$	
	$\xleftarrow{b}$	$b \xleftarrow{\$} \{0, 1, 2\}$.
If $b = 0$ set $z = (\underline{y}, \pi)$.		Accept if c_1 and c_2 are correct.
If $b = 1$ set $z = (\underline{y} + \underline{s}, \pi)$.	$\xrightarrow{z}$	Accept if c_1 and c_3 are correct.
If $b = 2$ set $z = (\pi(\underline{y}), \pi(\underline{s}))$.		Accept if c_2 and c_3 are correct and $\mathsf{wt}(\pi(\underline{s})) = w$.

It is easy to see that an honest prover is always accepted. In this case the protocol is said to be *complete*.

In terms of zero-knowledge, the scheme admits a simple proof. Very informally, the only data revealed during a run of the protocol is the two random objects $\underline{y}$ and π, the permuted strings $\pi(\underline{y})$ and $\pi(\underline{s})$ and the padding $\underline{y}+\underline{s}$. Clearly, $\underline{y}, \pi$ and $\pi(\underline{y})$ provide no information on s; the permuted string $\pi(\underline{s})$ doesn't leak anything apart from the weight of $\underline{s}$ (which is already known), while $\underline{y}+\underline{s}$ acts like a one-time pad. This is because $\underline{y}$ is randomly chosen, hence on average $\mathsf{wt}(\underline{y}) = n/2$ and this is large enough to mask the support of $\underline{s}$.

The biggest flaw of the scheme is that it is very easy for an impersonator to provide a forgery. More specifically, an impersonator would be able to reply correctly to two of the three challenges, arbitrarily, in the following way:

- The impersonator chooses random $\underline{y}$ and π plus another string $\underline{x} \in \mathbb{F}_2^n$ (not necessarily of low weight) such that $H\underline{x}^T = H\underline{y}^T + \underline{S}$, then builds c_1 and c_2 normally and $c_3 = \mathcal{H}(\pi(\underline{x}))$. It replies to the challenge with $(\underline{y}, \pi)$ if $b = 0$, or $(\underline{x}, \pi)$ if $b = 1$. The strategy fails for $b = 2$.

- The impersonator chooses random $\underline{y}$ and π plus another random string $\underline{x} \in \mathbb{W}_{2,n,w}$, then builds c_1 and c_2 normally and $c_3 = \mathcal{H}(\pi(\underline{y}+\underline{x}))$. It replies to the challenge with $(\underline{y}, \pi)$ if $b = 0$ or $(\pi(\underline{y}), \pi(\underline{x}))$ if $b = 2$. The strategy fails for $b = 1$.

- The impersonator chooses random $\underline{y}$ and π plus another random string $\underline{x} \in \mathbb{W}_{2,n,w}$, then builds $c_1 = \mathcal{H}(\pi, H(\underline{y}+\underline{x})^T + \underline{S})$, $c_2 = \mathcal{H}(\pi(\underline{y}))$ and $c_3 = \mathcal{H}(\pi(\underline{y}+\underline{x}))$. It replies to the challenge with $(\underline{y}+\underline{x}, \pi)$ if $b = 1$ or $(\pi(\underline{y}), \pi(\underline{x}))$ if $b = 2$. The strategy fails for $b = 0$.

Overall the probability of cheating is exactly 2/3. This means that an honest prover, in order to be accepted, needs to repeat the protocol many times. The author suggests 35 repetitions, leading to a cheating probability of $10^{-6} \approx 2^{-20}$, a weak authentication level. Still, even with this relatively small number of repetitions, communication costs per round amount to nearly 1146 bits for the original set of parameters $[512, 256, 56]$, for a total of more than 40110 bits. This results in a very long signature (> 150Kb) when the scheme is instantiated in the Fiat-Shamir protocol. Moreover, the proposed parameters are susceptible to general decoding attacks and the public key is very large, as for all code-based schemes. It is easy to imagine that, with parameters secure against modern criteria (e.g. at least 2^{128} security level for general decoding attacks and a minimum authentication level of 2^{-32}), the scheme would be even less practical.