Palestine Journal of Mathematics

Vol 14(2)(2025) , 158–165

MIXED DNA CRYPTOGRAPHY

N. Bennenni, K. Guenda, A. Delhoum and T. A. Gulliver

Communicated by Ayman Badawi

MSC 2010 Classifications: Primary 11T71; Secondary 94B05.

Keywords and phrases: Cryptography, DNA codes, Gray image, DNA cryptosystem.

Abstract In this paper, we introduce a new bijection mapping the elements of $\mathbb{F}_4R$ to DNA quadruplets $\{A, C, G, T\}^4$. This bijection serves as a powerful tool to avoid undesirable hybridization in DNA-based systems. It is used to develop a new cryptosystem based on the principle of hybridization that offers enhanced security and versatility. These results provide new avenues for cryptographic applications in biologically-inspired systems.

1 Introduction

Conventional cryptography is typically implemented on digital computing devices with silicon based storage. DNA cryptography employs strands of DNA that are manipulated using biological techniques [13]. DNA codes with conventional cryptography algorithms were considered in [7]. A symmetric key algorithm based on DNA strands, nucleotides, codon base pair rules, mutation, and conversion of DNA to mRNA was proposed in [9]. The security of the algorithm was improved using a Feistel inspired structure [10]. Cyclic DNA codes over the ring $\mathbb{F}_2[u]/(u^6)$ were constructed in [3] considering a one to one correspondence between the ring elements and the 64 codons over the alphabet $\{A, G, C, T\}$. In [1], linear simplex and MacDonald codes over mixed rings were presented and used to obtain secret-sharing schemes. The structure of cyclic codes over the ring $\mathbb{F}_q + u\mathbb{F}_q + u^2\mathbb{F}_q + \ldots + u^{k-1}\mathbb{F}_q$ was studied in [2].

Constraints on the GC-content have been used to avoid undesirable hybridization and to ensure similar melting temperatures [3, 5]. In this paper, we introduce a new bijection mapping the elements of $\mathbb{F}_4R$ to DNA quadruplets $\{A, C, G, T\}^4$. This bijection serves as a powerful tool for protecting against undesirable hybridization in DNA-based systems. It is used to obtain a new cryptosystem based on the principle of hybridization that offers enhanced security and versatility. This provides new possibilities for cryptographic applications in biologically-inspired systems.

The remainder of this paper is organized as follows. Section 2 presents some preliminary results for the ring $\mathbb{F}_4R$. A new bijection between the elements of $\mathbb{F}_4R$ and DNA quadruplets $\{A, G, C, T\}^4$ is given and a Gray map between $\mathbb{F}_4R$ and $\mathbb{F}_4^4$ is obtained. Then the new cryptosystem based on hybridization is presented in Section 3.

2 Notation and Preliminaries

Let $\mathbb{F}_4 = \{0, 1, w, w^2\}$ where $w^2 + w + 1 = 0$ and α be an even integer. The Hamming weight of the vector $x \in \mathbb{F}_4^\alpha$, denoted $w_H(x)$, is defined as the number of nonzero coordinates in x. The Hamming distance between two vectors $x, y \in \mathbb{F}_4^\alpha$ is given by $d_H(x, y) = w_H(x - y)$ and the minimum Hamming distance of a code C is defined as

$$d_H(C) = \{\min(d_H(x, y)) : x, y \in C, x \neq y\}.$$

Let θ be the Frobenius automorphism of $\mathbb{F}_4$. Denote $R = \mathbb{F}_4[v]/(v^3 - v) = \mathbb{F}_4 + v\mathbb{F}_4 + v^2\mathbb{F}_4$ with $v^3 = v$. This is a semi-local ring with ideals given by (0), $(1 + v + v^2)$, (v^2), $(1 + v^2)$, $(v+v^2)$, and the maximal ideals are (v) and $(1+v)$. We define the automorphism $\tilde{\theta}$ by extending

θ as follows

$$\begin{array}{rcl} \tilde{\theta} : \mathbb{F}_4 + v\mathbb{F}_4 + v^2\mathbb{F}_4 & \longrightarrow & \mathbb{F}_4 + v\mathbb{F}_4 + v^2\mathbb{F}_4 \\ r = a + bv + cv^2 & \longmapsto & \tilde{\theta}(r) = \theta(a) + v\theta(b) + v^2\theta(c) \end{array}$$

The order of $\tilde{\theta}$ is $|\tilde{\theta}| = 2$ and R is the fixed ring of $\tilde{\theta}$. A Gray map from R to $\mathbb{F}_4^3$ is given by

$$\begin{array}{rcl} \rho : R & \longrightarrow & \mathbb{F}_4^3 \\ r = a + bv + cv^2 & \longmapsto & \rho(r) = (a + c, a + b, c). \end{array}$$

Define $\mathbb{F}_4 R = \{(a, b + vc + dv^2) : a, b, c, d \in \mathbb{F}_4\}$ and consider the natural homomorphism

$$\Psi : R \to \mathbb{F}_4$$

$$b + vc + dv^2 \to b.$$

It can easily be verified that $\mathbb{F}_4 R$ is an R-module under multiplication given by

$$d * (a, b) = (\Psi(d)a, db) \text{ with } d \in R \text{ and } (a, b) \in \mathbb{F}_4 R. \tag{2.1}$$

This multiplication extends naturally to $\mathbb{F}_4^\alpha R^\beta$. Let $\mathbf{x} = (a_0, a_1, \ldots, a_{\alpha-1}, b_0, b_1, \ldots, b_{\beta-1}) \in \mathbb{F}_4^\alpha R^\beta$ and $d \in R$, and then

$$d * \mathbf{x} = (\Psi(d)a_0, \Psi(d)a_1, \ldots, \Psi(d)a_{\alpha-1}, db_0, db_1, \ldots, db_{\beta-1}). \tag{2.2}$$

DNA is formed by strands linked together and twisted in the shape of a double helix. Each strand is a sequence of four possible nucleotides, two purines, adenine A and guanine G, and two pyrimidines, thymine T and cytosine C. The ends of a DNA strand are chemically polar with $5'$ and $3'$ ends, which implies that the strands are oriented. Hybridization, known as base pairing, occurs when a strand binds to another strand to form a double strand of DNA. The strands are linked following the Watson-Crick model. Every A is linked with a T and every C with a G, and vice versa. We denote the complement of x by $\hat{x}$, i.e., $\hat{A} = T, \hat{T} = A, \hat{G} = C$ and $\hat{C} = G$. Since the cardinality of $\mathbb{F}_4 R$ is 256, we can construct a one-to-one correspondence between the elements of $\mathbb{F}_4 R$ and the 256 quadruplets over the alphabet $\{A, C, G, T\}$ by the map φ given in Table 1. It can easily be verified that for all $x \in \mathbb{F}_4 R$

$$x + \hat{x} = (w^2, v + v^2). \tag{2.3}$$

Table 1. The correspondence between elements of $\{A, G, C, T\}^4$ and elements of $\mathbb{F}_4 R$

r	DNA	r	DNA	r	DNA	r	DNA
$(0, 0)$	CAAA	$(0, 1)$	CTTA	$(0, w)$	CCCA	$(0, w^2)$	CGGA
$(0, v^2)$	CTAT	$(0, 1 + v^2)$	CATT	$(0, w + v^2)$	CGCT	$(0, w^2 + v^2)$	CCGT
$(0, wv^2)$	CCAC	$(0, 1 + wv^2)$	CGTC	$(0, w + wv^2)$	CACC	$(0, w^2 + wv^2)$	CTGC
$(0, w^2v^2)$	CGAG	$(0, 1 + w^2v^2)$	CCTG	$(0, w + w^2v^2)$	CTCG	$(0, w^2 + w^2v^2)$	CAGG
$(0, v)$	CATA	$(0, 1 + v)$	CTAA	$(0, w + v)$	CCGA	$(0, w^2 + v)$	CGCA
$(0, v + v^2)$	CTTT	$(0, 1 + v + v^2)$	CAAT	$(0, w + v + v^2)$	CGGT	$(0, w^2 + v + v^2)$	CCCT
$(0, v + wv^2)$	CCTC	$(0, 1 + v + wv^2)$	CGAC	$(0, w + v + wv)$	CAGC	$(0, w^2 + v + wv^2)$	CTCC
$(0, v + w^2v^2)$	CGTG	$(0, 1 + v + w^2v^2)$	CCAG	$(0, w + v + w^2v^2)$	CTGG	$(0, w^2 + v + w^2v^2)$	CACG
$(0, wv)$	CACA	$(0, 1 + wv)$	CTGA	$(0, w + wv)$	CCAA	$(0, w^2 + wv)$	CGTA
$(0, wv + v^2)$	CTCT	$(0, 1 + wv + v^2)$	CAGT	$(0, w + wv + v^2)$	CGAT	$(0, w^2 + wv + v^2)$	CCTT
$(0, wv + wv^2)$	CCCC	$(0, 1 + wv + wv^2)$	CGGC	$(0, w + wv + wv^2)$	CAAC	$(0, w^2 + wv + wv^2)$	CTTC
$(0, w + w^2v^2)$	CGCG	$(0, 1 + wv + w^2v^2)$	CCGG	$(0, w + wv + w^2v^2)$	CTAG	$(0, w^2 + wv + w^2v^2)$	CATG
$(0, w^2v$	CAGA	$(0, 1 + w^2v)$	CTCA	$(0, w + w^2v)$	CCTA	$(0, w^2 + w^2v)$	CGAA
$(0, w^2v + v^2)$	CTGT	$(0, 1 + w^2v + v^2)$	CACT	$(0, w + w^2v + v^2)$	CGTT	$(0, w^2 + w^2v + v^2)$	CCAT
$(0, w^2v + wv^2)$	CCGC	$(0, 1 + w^2v + wv^2)$	CGCC	$(0, w + w^2v + wv^2)$	CATC	$(0, w^2 + w^2v + wv^2)$	CTAC
$(0, w^2v + w^2v^2)$	CGGG	$(0, 1 + w^2v + w^2v^2)$	CCCG	$(0, w + w^2v + w^2v^2)$	CTTG	$(0, w^2 + w^2v + w^2v^2)$	CAAG
$(w^2, 0)$	GAAA	$(w^2, 1)$	GTTA	(w^2, w)	GCCA	(w^2, w^2)	GGGA
(w^2, v^2)	GTAT	$(w^2, 1 + v^2)$	GATT	$(w^2, w + v^2)$	GGCT	$(w^2, w^2 + v^2)$	GCGT
(w^2, wv^2)	GCAC	$(w^2, 1 + wv^2)$	GGTC	$(w^2, w + wv^2)$	GACC	$(w^2, w^2 + wv^2)$	GTGC
(w^2, w^2v^2)	GGAG	$(w^2, 1 + w^2v^2)$	GCTG	$(w^2, w + w^2v^2)$	GTCG	$(w^2, w^2 + w^2v^2)$	GAGG
(w^2, v)	GATA	$(w^2, 1 + v)$	GTAA	$(w^2, w + v)$	GCGA	$(w^2, w^2 + v)$	GGCA
$(w^2, v + v^2)$	GTTT	$(w^2, 1 + v + v^2)$	GAAT	$(w^2, w + v + v^2)$	GGGT	$(w^2, w^2 + v + v^2)$	GCCT
$(w^2, v + wv^2)$	GCTC	$(w^2, 1 + v + wv^2)$	GGAC	$(w^2, w + v + wv)$	GAGC	$(w^2, w^2 + v + wv^2)$	GTCC
$(w^2, v + w^2v^2)$	GGTG	$(w^2, 1 + v + w^2v^2)$	GCAG	$(w^2, w + v + w^2v^2)$	GTGG	$(w^2, w^2 + v + w^2v^2)$	GACG
(w^2, wv)	GACA	$(w^2, 1 + wv)$	GTGA	$(w^2, w + wv)$	GCAA	$(w^2, w^2 + wv)$	GGTA
$(w^2, wv + v^2)$	GTCT	$(w^2, 1 + wv + v^2)$	GAGT	$(w^2, w + wv + v^2)$	GGAT	$(w^2, w^2 + wv + v^2)$	GCTT
$(w^2, wv + wv^2)$	GCCC	$(w^2, 1 + wv + wv^2)$	GGGC	$(w^2, w + wv + wv^2)$	GAAC	$(w^2, w^2 + wv + wv^2)$	GTTC
$(w^2, w + w^2v^2)$	GGCG	$(w^2, 1 + wv + w^2v^2)$	GCGG	$(w^2, w + wv + w^2v^2)$	GTAG	$(w^2, w^2 + wv + w^2v^2)$	GATG
(w^2, w^2v)	GAGA	$(w^2, 1 + w^2v)$	GTCA	$(w^2, w + w^2v)$	GCTA	$(w^2, w^2 + w^2v)$	GGAA
$(w^2, w^2v + v^2)$	GTGT	$(w^2, 1 + w^2v + v^2)$	GACT	$(w^2, w + w^2v + v^2)$	GGTT	$(w^2, w^2 + w^2v + v^2)$	GCAT
$(w^2, w^2v + wv^2)$	GCGC	$(w^2, 1 + w^2v + wv^2)$	GGCC	$(w^2, w + w^2v + wv^2)$	GATC	$(w^2, w^2 + w^2v + wv^2)$	GTAC
$(w^2, w^2v + w^2v^2)$	GGGG	$(w^2, 1 + w^2v + w^2v^2)$	GCCG	$(w^2, w + w^2v + w^2v^2)$	GTTG	$(w^2, w^2 + w^2v + w^2v^2)$	GAAG
$(w, 0)$	TAAA	$(w, 1)$	TTTA	(w, w)	TCCA	(w, w^2)	TGGA
(w, v^2)	TTAT	$(w, 1 + v^2)$	TATT	$(w, w + v^2)$	TGCT	$(w, w^2 + v^2)$	TCGT
(w, wv^2)	TCAC	$(w, 1 + wv^2)$	TGTC	$(w, w + wv^2)$	TACC	$(w, w^2 + wv^2)$	TTGC
(w, w^2v^2)	TGAG	$(w, 1 + w^2v^2)$	TCTG	$(w, w + w^2v^2)$	TTCG	$(w, w^2 + w^2v^2)$	TAGG
(w, v)	TATA	$(w, 1 + v)$	TTAA	$(w, w + v)$	TCGA	$(w, w^2 + v)$	TGCA
$(w, v + v^2)$	TTTT	$(w, 1 + v + v^2)$	TAAT	$(w, w + v + v^2)$	TGGT	$(w, w^2 + v + v^2)$	TCCT
$(w, v + wv^2)$	TCTC	$(w, 1 + v + wv^2)$	TGAC	$(w, w + v + wv)$	TAGC	$(w, w^2 + v + wv^2)$	TTCC
$(w, v + w^2v^2)$	TGTG	$(w, 1 + v + w^2v^2)$	TCAG	$(w, w + v + w^2v^2)$	TTGG	$(w, w^2 + v + w^2v^2)$	TACG
(w, wv)	TACA	$(w, 1 + wv)$	TTGA	$(w, w + wv)$	TCAA	$(w, w^2 + wv)$	TGTA
$(w, wv + v^2)$	TTCT	$(w, 1 + wv + v^2)$	TAGT	$(w, w + wv + v^2)$	TGAT	$(w, w^2 + wv + v^2)$	TCTT
$(w, wv + wv^2)$	TCCC	$(w, 1 + wv + wv^2)$	TGGC	$(w, w + wv + wv^2)$	TAAC	$(w, w^2 + wv + wv^2)$	TTTC
$(w, w + w^2v^2)$	TGCG	$(w, 1 + wv + w^2v^2)$	TCGG	$(w, w + wv + w^2v^2)$	TTAG	$(w, w^2 + wv + w^2v^2)$	TATG
(w, w^2v)	TAGA	$(w, 1 + w^2v)$	TTCA	$(w, w + w^2v)$	TCTA	$(w, w^2 + w^2v)$	TGAA
$(w, w^2v + v^2)$	TTGT	$(w, 1 + w^2v + v^2)$	TACT	$(w, w + w^2v + v^2)$	TGTT	$(w, w^2 + w^2v + v^2)$	TCAT
$(w, w^2v + wv^2)$	TCGC	$(w, 1 + w^2v + wv^2)$	TGCC	$(w, w + w^2v + wv^2)$	TATC	$(w, w^2 + w^2v + wv^2)$	TTAC
$(w, w^2v + w^2v^2$	TGGG	$(w, 1 + w^2v + w^2v^2)$	TCCG	$(w, w + w^2v + w^2v^2)$	TTTG	$(w, w^2 + w^2v + w^2v^2)$	TAAG
$(1, 0)$	AAAA	$(1, 1)$	ATTA	$(1, w)$	ACCA	$(1, w^2)$	AGGA
$(1, v^2)$	ATAT	$(1, 1 + v^2)$	AATT	$(1, w + v^2)$	AGCT	$(1, w^2 + v^2)$	ACGT
$(1, wv^2)$	ACAC	$(1, 1 + wv^2)$	AGTC	$(1, w + wv^2)$	AACC	$(1, w^2 + wv^2)$	ATGC
$(1, w^2v^2)$	AGAG	$(1, 1 + w^2v^2)$	ACTG	$(1, w + w^2v^2)$	ATCG	$(1, w^2 + w^2v^2)$	AAGG
$(1, v)$	AATA	$(1, 1 + v)$	ATAA	$(1, w + v)$	ACGA	$(1, w^2 + v)$	AGCA
$(1, v + v^2)$	ATTT	$(1, 1 + v + v^2)$	AAAT	$(1, w + v + v^2)$	AGGT	$(1, w^2 + v + v^2)$	ACCT
$(1, v + wv^2)$	ACTC	$(1, 1 + v + wv^2)$	AGAC	$(1, w + v + wv)$	AAGC	$(1, w^2 + v + wv^2)$	ATCC
$(1, v + w^2v^2)$	AGTG	$(1, 1 + v + w^2v^2)$	ACAG	$(1, w + v + w^2v^2)$	ATGG	$(1, w^2 + v + w^2v^2)$	AACG
$(1, wv)$	AACA	$(1, 1 + wv)$	ATGA	$(1, w + wv)$	ACAA	$(1, w^2 + wv)$	AGTA
$(1, wv + v^2)$	ATCT	$(1, 1 + wv + v^2)$	AAGT	$(1, w + wv + v^2)$	AGAT	$(1, w^2 + wv + v^2)$	ACTT
$(1, wv + wv^2)$	ACCC	$(1, 1 + wv + wv^2)$	AGGC	$(1, w + wv + wv^2)$	AAAC	$(1, w^2 + wv + wv^2)$	ATTC
$(1, w + w^2v^2)$	AGCG	$(1, 1 + wv + w^2v^2)$	ACGG	$(1, w + wv + w^2v^2)$	ATAG	$(1, w^2 + wv + w^2v^2)$	AATG
$(1, w^2v)$	AAGA	$(1, 1 + w^2v)$	ATCA	$(1, w + w^2v)$	ACTA	$(1, w^2 + w^2v)$	AGAA
$(1, w^2v + v^2)$	ATGT	$(1, 1 + w^2v + v^2)$	AACT	$(1, w + w^2v + v^2)$	AGTT	$(1, w^2 + w^2v + v^2)$	ACAT
$(1, w^2v + wv^2)$	ACGC	$(1, 1 + w^2v + wv^2)$	AGCC	$(1, w + w^2v + wv^2)$	AATC	$(1, w^2 + w^2v + wv^2)$	ATAC
$(1, w^2v + w^2v^2)$	AGGG	$(1, 1 + w^2v + w^2v^2)$	ACCG	$(1, w + w^2v + w^2v^2)$	ATTG	$(1, w^2 + w^2v + w^2v^2)$	AAAG

2.1 Gray Image of DNA Codes

Now, we give a Gray map which translates the properties of DNA codes to the binary case. Define the Gray weight of $r \in R$ as $wt_G(r) = wt_H(\rho(r))$ and consider the extension to R^β

$$x = (x_1, \ldots, x_\beta) \mapsto \rho(x) = (a_1 + c_1, \ldots, a_\beta + c_\beta, a_1 + b_1, \ldots, a_\beta + b_\beta, c_1, \ldots, c_\beta) \in \mathbb{F}_4^{3\beta}.$$

Then the Gray weight of an element $x \in R^\beta$ is $wt_G(x) = \sum_{i=1}^{\beta} wt_G(x_i)$. For elements $x, y \in R^\beta$, the Gray distance between x and y is $d_G(x, y) = wt_G(x - y)$, and the minimum Gray distance of a code C is $d_G(C) = \{\min(d_G(x, y)) : x, y \in C, x \neq y\}$. Clearly, ρ is $\mathbb{F}_4$-linear and is a bijection map from R^β to $\mathbb{F}_4^{3\beta}$.

Using the map ρ, define the Gray map

$$\begin{array}{ccc} \phi : \mathbb{F}_4^{\alpha} R^{\beta} & \longrightarrow & \mathbb{F}_4^{\alpha+3\beta} \\ (x, y) & \longmapsto & (x, \rho(y)). \end{array}$$

The corresponding Gray weight of the element $(x, y) \in \mathbb{F}_4^{\alpha} R^{\beta}$ is defined as $wt_G(x, y) = wt_H(x) + wt_G(y)$ and the Gray distance between $c, c' \in F_4^{\alpha} R^{\beta}$ as $d_G(c, c') = wt_G(c - c')$. The binary map μ is an $\mathbb{F}_2$-linear bijective map from $\mathbb{F}_4^n$ to $\mathbb{F}_2^{2n}$.

Let $x = (a_1 + wb_1, \ldots, a_n + wb_n) \in \mathbb{F}_4^n$, where $a_i, b_i \in \mathbb{F}_2$, $i = 1, \ldots, n$, and $w^2+w+1 = 0$. The binary image $\mu(x)$ of x is then $\mu(x) = (a_1, b_1, \ldots, a_n, b_n) \in \mathbb{F}_2^{2n}$. Table 2 gives the Gray image of the DNA quadruplets in Table 1.

Table 2. Gray images of the DNA quadruplets in Table 1

r	$\phi(r)$	r	$\phi(r)$	r	$\phi(r)$	r	$\phi(r)$
$(0, 0, 0, 0)$	CAAA	$(0, 1, 1, 0)$	CTTA	$(0, w, w, 0)$	CCCA	$(0, w^2, w^2, 0)$	CGGA
$(0, 1, 0, 1)$	CTAT	$(0, 0, 1, 1)$	CATT	$(0, w^2, w, 1)$	CGCT	$(0, w, w^2, 1)$	CCGT
$(0, w, 0, w)$	CCAC	$(0, w^2, 1, w)$	CGTC	$(0, 0, w, w)$	CACC	$(0, 1, w^2, w)$	CTGC
$(0, w^2, 0, w^2)$	CGAG	$(0, w, 1, w^2)$	CCTG	$(0, 1, w, w^2)$	CTCG	$(0, 0, w^2, w^2)$	CAGG
$(0, 0, 1, 0)$	CATA	$(0, 1, 0, 0)$	CTAA	$(0, w, w^2, 0)$	CCGA	$(0, w^2, w, 0)$	CGCA
$(0, 1, 1, 1)$	CTTT	$(0, 0, 0, 1)$	CAAT	$(0, w^2, w^2, 1)$	CGGT	$(0, w, w, 1)$	CCCT
$(0, w, 1, w)$	CCTC	$(0, w^2, 0, w)$	CGAC	$(0, 0, w^2, w)$	CAGC	$(0, 1, w, w)$	CTCC
$(0, w^2, 1, w^2)$	CGTG	$(0, w, 0, w^2)$	CCAG	$(0, 1, w^2, w^2)$	CTGG	$(0, 0, w, w^2)$	CACG
$(0, 0, w, 0)$	CACA	$(0, 1, w^2, 0)$	CTGA	$(0, w, 0, 0)$	CCAA	$(0, w^2, 1, 0)$	CGTA
$(0, 1, w, 1)$	CTCT	$(0, 0, w^2, 1)$	CAGT	$(0, w^2, 0, 1)$	CGAT	$(0, w, 1, 1)$	CCTT
$(0, w, w, w)$	CCCC	$(0, w^2, w^2, w)$	CGGC	$(0, 0, 0, w)$	CAAC	$(0, 1, 1, w)$	CTTC
$(0, 1, w, w^2)$	CGCG	$(0, w, w^2, w^2)$	CCGG	$(0, 1, 0, w^2)$	CTAG	$(0, 0, 1, w^2)$	CATG
$(0, 0, w^2, 0)$	CAGA	$(0, 1, w, 0)$	CTCA	$(0, w, 1, 0)$	CCTA	$(0, w^2, 0, 0)$	CGAA
$(0, 1, w^2, 1)$	CTGT	$(0, 0, w, 1)$	CACT	$(0, w^2, 1, 1)$	CGTT	$(0, w, 0, 1)$	CCAT
$(0w, w^2, w)$	CCGC	$(0, w^2, w, w)$	CGCC	$(0, 0, 1, w)$	CATC	$(0, 1, 0, w)$	CTAC
$(0, w^2, w^2, w^2)$	CGGG	$(0, w, w, w^2)$	CCCG	$(0, 1, 1, w^2)$	CTTG	$(0, 0, 0, w^2)$	CAAG
$(w^2, 0, 0, 0)$	GAAA	$(w^2, 1, 0, 0)$	GTTA	$(w^2, w, w, 0)$	GCCA	$(w^2, w^2, w^2, 0)$	GGGA
$(w^2, 1, 0, 1)$	GTAT	$(w^2, 0, 1, 1)$	GATT	$(w^2, w^2, w, 1)$	GGCT	$(w^2, w, w^2, 1)$	GCGT
$(w^2, w, 0, w)$	GCAC	$(w^2, w^2, 1, w)$	GGTC	$(w^2, 0, w, w)$	GACC	$(w^2, 1, w^2, w)$	GTGC
$(w^2, w^2, 0, w^2)$	GGAG	$(w^2, w, 1, w^2)$	GCTG	$(w^2, 1, w, w^2)$	GTCG	$(w^2, 0, w^2, w^2)$	GAGG
$(w^2, 0, 1, 0)$	GATA	$(w^2, 1, 0, 0)$	GTAA	$(w^2, w, w^2, 0)$	GCGA	$(w^2, w^2, w, 0)$	GGCA
$(w^2, 1, 1, 1)$	CTTT	$(w^2, 0, 0, 1)$	GAAT	$(w^2, w^2, w^2, 1)$	GGGT	$(w^2, w, w, 1)$	GCCT
$(w^2, w, 1, w)$	GCTC	$(w^2, w^2, 0, w)$	GGAC	$(w^2, 0, w^2, w)$	GAGC	$(w^2, 1, w, w)$	GTCC
$(w^2, w^2, 1, w^2)$	GGTG	$(w^2, w, 0, w^2)$	GCAG	$(w^2, 1, w^2, w^2)$	GTGG	$(w^2, 0, w, w^2)$	GACG
$(w^2, 0, w, 0)$	GACA	$(w^2, 1, w^2, 0)$	GTGA	$(w^2, w, 0, 0)$	GCAA	$(w^2, w^2, 1, 0)$	GGTA
$(w^2, 1, w, 1)$	GTCT	$(w^2, 0, w^2, 1)$	GAGT	$(w^2, w^2, 0, 1)$	GGAT	$(w^2, w, 1, 1)$	GCTT
(w^2, w, w, w)	GCCC	(w^2, w^2, w^2, w)	GGGC	$(w^2, 0, 0, w)$	GAAC	$(w^2, 1, 1, w)$	GTTC
$(w^2, 1, w, w^2)$	GGCG	(w^2, w, w^2, w^2)	GCGG	$(w^2, 1, 0, w^2)$	GTAG	$(w^2, 0, 1, w^2)$	GATG
$(w^2, 0, w^2, 0)$	GAGA	$(w^2, 1, w, 0)$	GTCA	$(w^2, w, 1, 0)$	GCTA	$(w^2, w^2, 0, 0)$	GGAA
$(w^2, 1, w^2, 1)$	GTGT	$(w^2, 0, w, 1)$	GACT	$(w^2, w^2, 1, 1)$	GGTT	$(w^2, w, 0, 1)$	GCAT
(w^2, w, w^2, w)	GCGC	(w^2, w^2, w, w)	GGCC	$(w^2, 0, 1, w)$	GATC	$(w^2, 1, 0, w)$	GTAC
(w^2, w^2, w^2, w^2)	GGGG	(w^2, w, w, w^2)	GCCG	$(w^2, 1, 1, w^2)$	GTTG	$(w^2, 0, 0, w^2)$	GAAG
$(w, 0, 0, 0)$	TAAA	$(w, 1, 0, 0)$	TTTA	$(w, w, w, 0)$	TCCA	$(w, w^2, w^2, 0)$	TGGA
$(w, 1, 0, 1)$	TTAT	$(w, 0, 1, 1)$	TATT	$(w, w^2, w, 1)$	TGCT	$(w, w, w^2, 1)$	TCGT
$(w, w, 0, w)$	TCAC	$(w, w^2, 1, w)$	TGTC	$(w, 0, w, w)$	TACC	$(w, 1, w^2, w)$	TTGC
$(w, w^2, 0, w^2)$	TGAG	$(w, w, 1, w^2)$	TCTG	$(w, 1, w, w^2)$	TTCG	$(w, 0, w^2, w^2)$	TAGG
$(w, 0, 1, 0)$	TATA	$(w, 1, 0, 0)$	TTAA	$(w, w, w^2, 0)$	TCGA	$(w, w^2, w, 0)$	TGCA
$(w, 1, 1, 1)$	CTTT	$(w, 0, 0, 1)$	TAAT	$(w, w^2, w^2, 1)$	TGGT	$(w, w, w, 1)$	TCCT
$(w, w, 1, w)$	TCTC	$(w, w^2, 0, w)$	TGAC	$(w, 0, w^2, w)$	TAGC	$(w, 1, w, w)$	TTCC
$(w, w^2, 1, w^2)$	TGTG	$(w, w, 0, w^2)$	TCAG	$(w, 1, w^2, w^2)$	TTGG	$(w, 0, w, w^2)$	TACG
$(w, 0, w, 0)$	TACA	$(w, 1, w^2, 0)$	TTGA	$(w, w, 0, 0)$	TCAA	$(w, w^2, 1, 0)$	TGTA
$(w, 1, w, 1)$	TTCT	$(w, 0, w^2, 1)$	TAGT	$(w, w^2, 0, 1)$	TGAT	$(w, w, 1, 1)$	TCTT
(w, w, w, w)	TCCC	(w, w^2, w^2, w)	TGGC	$(w, 0, 0, w)$	TAAC	$(w, 1, 1, w)$	TTTC
$(w, 1, w, w^2)$	TGCG	(w, w, w^2, w^2)	TCGG	$(w, 1, 0, w^2)$	TTAG	$(w, 0, 1, w^2)$	TATG
$(w, 0, w^2, 0)$	TAGA	$(w, 1, w, 0)$	TTCA	$(w, w, 1, 0)$	TCTA	$(w, w^2, 0, 0)$	TGAA
$(w, 1, w^2, 1)$	TTGT	$(w, 0, w, 1)$	TACT	$(w, w^2, 1, 1)$	TGTT	$(w, w, 0, 1)$	TCAT
(w, w, w^2, w)	TCGC	(w, w^2, w, w)	TGCC	$(w, 0, 1, w)$	TATC	$(w, 1, 0, w)$	TTAC
(w, w^2, w^2, w^2)	TGGG	(w, w, w, w^2)	TCCG	$(w, 1, 1, w^2)$	TTTG	$(w, 0, 0, w^2)$	TAAG
$(1, 0, 0, 0)$	AAAA	$(1, 1, 0, 0)$	ATTA	$(1, w, w, 0)$	ACCA	$(1, w^2, w^2, 0)$	AGGA
$(1, 1, 0, 1)$	ATAT	$(1, 0, 1, 1)$	AATT	$(1, w^2, w, 1)$	AGCT	$(1, w, w^2, 1)$	ACGT
$(1, w, 0, w)$	ACAC	$(1, w^2, 1, w)$	AGTC	$(1, 0, w, w)$	AACC	$(1, 1, w^2, w)$	ATGC
$(1, w^2, 0, w^2)$	AGAG	$(1, w, 1, w^2)$	ACTG	$(1, 1, w, w^2)$	ATCG	$(1, 0, w^2, w^2)$	AAGG
$(1, 0, 1, 0)$	AATA	$(1, 1, 0, 0)$	ATAA	$(1, w, w^2, 0)$	ACGA	$(1, w^2, w, 0)$	AGCA
$(1, 1, 1, 1)$	ATTT	$(1, 0, 0, 1)$	AAAT	$(1, w^2, w^2, 1)$	AGGT	$(1, w, w, 1)$	ACCT
$(1, w, 1, w)$	ACTC	$(1, w^2, 0, w)$	AGAC	$(1, 0, w^2, w)$	AAGC	$(1, 1, w, w)$	ATCC
$(1, w^2, 1, w^2)$	AGTG	$(1, w, 0, w^2)$	ACAG	$(1, 1, w^2, w^2)$	ATGG	$(1, 0, w, w^2)$	AACG
$(1, 0, w, 0)$	AACA	$(1, 1, w^2, 0)$	ATGA	$(1, w, 0, 0)$	ACAA	$(1, w^2, 1, 0)$	AGTA
$(1, 1, w, 1)$	ATCT	$(1, 0, w^2, 1)$	AAGT	$(1, w^2, 0, 1)$	AGAT	$(1, w, 1, 1)$	ACTT
$(1, w, w, w)$	ACCC	$(1, w^2, w^2, w)$	AGGC	$(1, 0, 0, w)$	AAAC	$(1, 1, 1, w)$	ATTC
$(1, 1, w, w^2)$	AGCG	$(1, w, w^2, w^2)$	ACGG	$(1, 1, 0, w^2)$	ATAG	$(1, 0, 1, w^2)$	AATG
$(1, 0, w^2, 0)$	AAGA	$(1, 1, w, 0)$	ATCA	$(1, w, 1, 0)$	ACTA	$(1, w^2, 0, 0)$	AGAA
$(1, 1, w^2, 1)$	ATGT	$(1, 0, w, 1)$	AACT	$(1, w^2, 1, 1)$	AGTT	$(1, w, 0, 1)$	ACAT
$(1, w, w^2, w)$	ACGC	$(1, w^2, w, w)$	AGCC	$(1, 0, 1, w)$	AATC	$(1, 1, 0, w)$	ATAC
$(1, w^2, w^2, w^2)$	AGGG	$(1, w, w, w^2)$	ACCG	$(1, 1, 1, w^2)$	ATTG	$(1, 0, 0, w^2)$	AAAG

The relationship between the binary image, the elements of the finite field $\mathbb{F}_4$ and the DNA quadruplets is shown in the following diagram

$$\begin{array}{ccc} \mathbb{F}_4 R & \xrightarrow{\phi} & \mathbb{F}_4^3 \\ \downarrow{\scriptstyle \varphi} & & \downarrow{\scriptstyle \mu} \\ DNA & \xrightarrow{\mu\circ\phi\circ\varphi^{-1}} & \mathbb{F}_2^6 \end{array}$$

3 A New DNA Cryptosystem

Hybridization refers to the process of combining complementary strands so each strand finds its Watson-Crick complement to form a DNA double helix [12]. We define a new map for hybridization between two DNA sequences x and y as follows

$$\begin{aligned} \mathbb{H} : (\text{DNA})^2 &\longrightarrow \mathbb{F}_2 \\ (x, y) &\longmapsto \mathbb{H}(x, y) = \begin{cases} 1, x = y^c \\ 0, x \neq y^c \end{cases} \end{aligned}$$

The proposed symmetric key algorithm employs Diffie-Hellman (DH) key exchange, Rivest-Shamir-Adleman (RSA) encryption, XOR and XNOR operations, the mixed automorphism $(\theta, \tilde{\theta})$ and the hybridization map [8, 4]. The keys generated using hybridization are transmitted through a secure channel using the DH and RSA protocols to ensure security. The encryption and decryption algorithms are given in Algorithms 1 and 2, respectively.

Algorithm 1 DNA Encryption

Input: Binary message m of length l.
Encryption

1: Map the message m using $\mu^{-1} \circ \phi^{-1}$ to an elements of $\mathbb{F}_4 R$ to obtain m'.
2: Apply the mixed automorphism $(\theta, \tilde{\theta})$ to m' to obtain m''.
3: Use Table 1 to convert m'' to an ssDNA sequence.
4: Use the composition map $(\mu \circ \phi \circ \varphi^{-1})$ to obtain the binary image of the ssDNA sequence.
5: Use the DH and RSA protocols to create a secret key (K, K') with the same length as the binary image of the ssDNA sequence.
6: XNOR the sequences from Steps 4 and 5 and transform the result into a DNA sequence.
7: Hybridize the DNA sequence from Step 3 and the DNA sequencefrom Step 6 using the map $\mathbb{H}$.
8: XOR the results of Steps 5 and 7.
9: Convert the result to a binary sequence and use the composition map $(\mu \circ \phi \circ \varphi^{-1})$ to obtain the ciphertext.

Algorithm 2 DNA Decryption

Input: The encrypted message and secret key (K, K') for the DH and RSA protocols.
Decryption

1: Convert the received DNA sequence to a binary sequence using the composition map $\mu \circ \phi \circ \varphi^{-1}$.
2: Concatenate copies of the secret key to obtain the same length as the sequence in Step 1.
3: XOR the sequences from Steps 1 and 2.
4: XNOR the sequences from Steps 2 and 3.
5: Convert the sequence obtained in Step 5 to a DNA sequence and then to the message using the composition map $(\varphi \circ \phi^{-1} \circ \mu^{-1})$.

3.1 Security Analysis

The security of the proposed cryptosystem depends on several factors. First, the mixed automorphism in Step 2 of Algorithm 1 conceals the structure of the initial single-stranded DNA (ssDNA) sequence. Second, the combined security of the DH and RSA protocols in Step 4. The DH protocol ensures perfect forward secrecy but the RSA protocol is vulnerable to quantum and Nittaj attacks [11]. However, combining these protocols to generate the secret key makes the algorithm robust by enhancing the security of the DH protocol to prevent man in the middle attacks [6]. Further, the hybridization used in Steps 5 and 6 ensures the integrity of the message. The probability of a successful brute-force attack on the algorithm is $\frac{1}{l^{8\times n\times l}}$, where l is the message length in bytes and n is the oligonucleotide ladder, e.g. the ss10 DNA ladder is an oligo length standard, i.e. ssDNA ladder, ranging from 10 to 100 nucleotides in 10-nucleotide increments [14]. For a 5- or 4-mer oligonucleotide ladder message, this probability is 1.94×10^{-84}.

Example 3.1. DNA Encryption
Consider the message $m = S_1 = 01000011, 01101111, 01100100, 01100101$. The corresponding quadruplets are obtained using the composition map $(\varphi \circ \phi^{-1} \circ \mu^{-1})$

$$S_2 = GTAT, TCAC, TGCT, AGGA,$$

$$S_4 = w^2, v^2, w, wv^2, w, w + wv^2, 1, w^2.$$

The mixed automorphism $(\theta, \tilde{\theta})$ is used to obtain

$$S_5 = w, v^2, w^2, w^2v^2, w^2, w^2 + v^2, 1, w,$$

which is

$$S_6 = TTAT, GGAG, GCGT, ACCA.$$

The Gray map gives

$$S_7 = w, 1, 0, 1, w^2, w^2, 0, w^2, w^2, w, w^2, 1, 1, w, w, 0$$

and the binary image of the message is

$$S_8 = 01100010, 11110011, 11011110, 10010100.$$

The secret key (K, K') generated using the DH and RSA protocols is

$$S_9 = 00001110, 00001110, 00000010, 00000010.$$

XNOR the secret key S_9 with the binary image of the message S_8 to obtain the binary DNA key

$$S_{10} = 10010011, 00000010, 00100011, 01101001.$$

Use the composition map $(\varphi \circ \phi^{-1} \circ \mu^{-1})$ to obtain the mixed secret key (K, K')

$$S_{11} = CAGT, CAGT, CAAT, CAAT.$$

The binary hybridization sequence is obtained by hybridizing S_{11} with S_6 to obtain

$$S_{12} = 00010000, 01000000, 01000000, 00000001.$$

XOR the binary DNA key S_{10} and the hybridization sequence S_{12} to get the encrypted binary message

$$S_{13} = 10000011, 01000010, 01100011, 01101000.$$

The encrypted DNA message is obtained using the composition map $(\varphi \circ \phi^{-1} \circ \mu^{-1})$.

$$S_{14} = AAAG, TAAT, TTAG, TTTC.$$

DNA Decryption
Convert the DNA sequence

$$S_{14} = AAAG, TAAT, TTAG, TTTC$$

to a binary sequence using the composition map $(\mu \circ \phi \circ \varphi - 1)$

$$S_{14} = 1000011, 01000010, 01100011, 01101001.$$

XOR the binary hybridizing sequence S_{12} and the sequence S_{14} to obtain

$$S_{15} = 10010011, 00000010, 00100011, 01101000.$$

XNOR the sequence S_{15} and the binary secret key sequence S_{14} to get

$$S_{16} = 01100010, 11110011, 11011110, 10010101.$$

Use the composition map $(\varphi \circ \phi^{-1} \circ \mu^{-1})$ and the mixed automorphism $(\theta, \tilde{\theta})$ to get the DNA message

$$S_2 = GTAT, TCAC, TGCT, AGGA.$$

References

[1] K. Chatouh, *Linear codes over* $\mathfrak{R} = \mathbb{Z}_7(\mathbb{Z}_7 + u\mathbb{Z}_7)(\mathbb{Z}_7 + v\mathbb{Z}_7 + w\mathbb{Z}_7 + vw\mathbb{Z}_7)$, Palest. J. Math., **13(III)**, 55–62, 2024.

[2] M. M. Al-Ashker and J. Chen, *Cyclic codes of arbitrary length over* $F_q + uF_q + u^2F_q + \ldots + u^{k-1}F_q$, Palest. J. Math., **2(1)**, 72–80, 2013.

[3] N. Bennenni, K. Guenda, and S. Mesnager, *New DNA cyclic codes over rings*, Adv. Math. Commun., **11(1)**, 83–98, 2017.

[4] G. R. Blakley and I. Borosh, *Rivest-Shamir-Adleman public key cryptosystems do not always conceal messages*, Comput. Math. Appl., **5(3)**, 169–178, 1979.

[5] A. G. Frutos, Q. Liu, A. J. Thiel, A. M. W. Sanner, A. E. Condon, L. M. Smith, and R. M. Corn, *Demonstration of a word design strategy for DNA computing on surfaces*, Nucleic Acids Res., **25(23)**, 4748–4757, 1997.

[6] C. Gupta and N. V. S. Reddy, *Enhancement of security of Diffie-Hellman key exchange protocol using RSA cryptography*, J. Phys. Conf. Ser., **2161**, 012014, 2022

[7] B. T. Hammad, A. M. Sagheer, I. T. Ahmed, and N. Jamil, *A comparative review on symmetric and asymmetric DNA-based cryptography*, Bull. Electr. Eng. Inform., **9(6)**, 2484–2491, 2020.

[8] M. Hellman, *New directions in cryptography*, IEEE Trans. Inform. Theory, **22(6)**, 644–654, 1976.

[9] M. Karimi and W. Haider, *Cryptography using DNA nucleotides*, Int. J. Comput. Appl., **168(7)**, 16–18, 2017.

[10] A. K. Kaundal and A. Verma, *Extending Feistel structure to DNA cryptography*, J. Discrete Math. Sci. Cryptogr., **18(4)**, 349–362, 2015.

[11] A. Nitaj, M. R. K. Ariffin, D. I. Nassr, and H. M. Bahig, *New attacks on the RSA cryptosystem*, in D. Pointcheval and D. Vergnaud (eds), Progress in Cryptology – Proc. AFRICACRYPT, Lecture Notes in Computer Science, **8469**, 178–198, Springer, Cham, Switzerland.

[12] O. Milenkovic and N. Kashyap, *On the design of codes for DNA computing*, in Ø. Ytrehus (ed.), Coding and Cryptography - Proc. WCC, Lecture Notes in Computer Science, **3969**, 100–119, Springer, Berlin, Germany, 2005.

[13] C. Popovici, *Aspects of DNA cryptography*, Ann. Univ. Craiova Math. Comput. Sci. Ser., **37(3)**, 147–151, 2010.

[14] C.-H. Tung, J. Wang, M. J. Leibowitz, and S. Stein, *Dual-specificity interaction of HIV-1 TAR RNA with Tat peptide-oligonucleotide conjugates*, Bioconjugate Chem., **6(3)**, 292–295, 1995.

Author information

N. Bennenni, Department of Algebra and Number Theory, Faculty of Mathematics, University of Science and Technology Houari Boumediene USTHB, Bab Ezzouar, Algeria.
E-mail: nabil.bennenni@gmail.com

K. Guenda, Department of Algebra and Number Theory, Faculty of Mathematics, University of Science and Technology Houari Boumediene USTHB, Bab Ezzouar, Algeria.
E-mail: ken.guenda@gmail.com

A. Delhoum, Department of Algebra and Number Theory, Faculty of Mathematics, University of Science and Technology Houari Boumediene USTHB, Bab Ezzouar, Algeria.
E-mail: aminadelm@gmail.com

T. A. Gulliver, Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada.
E-mail: agullive@ece.uvic.ca

Received: 2024-04-23
Accepted: 2024-11-20