Άσκηση 1 (2.2)

Το κρυπτογραφημένο μήνυμα:

¨οκηθμφδζθγοθχυκχσφθμφμχγ'

Μετατροπή σε αριθμητική μορφή:

$15, 10, 7, 8, 12, 21, 4, 6, 8, 3, 15, 8, 22, 20, 10, 22, 18, 21, 8, 12, 21, 12, 22, 3$

Εύρεση της ρίζας x_0 του τριωνύμου

Δίνεται το πολυώνυμο:

$$g(x) = x^2 + 3x + 1$$

Η ρίζα του τριωνύμου βρίσκεται λύνοντας:

$$x^2 + 3x + 1 = 0$$

Χρησιμοποιούμε Διακρίνουσα:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Όπου: $a = 1$, $b = 3$, $c = 1$

Υπολογίζουμε τη διακρίνουσα:

$$\Delta = 3^2 - 4(1)(1) = 9 - 4 = 5$$

Οι ρίζες είναι:

$$x_0 = \frac{-3 \pm \sqrt{5}}{2}$$

Παίρνουμε τη θετική ρίζα:

$$x_0 = \frac{-3 + \sqrt{5}}{2}$$

Υπολογισμός του $f(x_0)$

Δίνεται το πολυώνυμο:

$$f(x) = x^5 + 3x^3 + 7x^2 + 3x^4 + 5x + 4$$

Αντικαθιστούμε $x = x_0$ και χρησιμοποιούμε τη σχέση:

$$x_0^2 = -3x_0 - 1$$

για να απλοποιήσουμε τις εκφράσεις.

Με τους υπολογισμούς καταλήγουμε:

$$f(x_0) = 3$$

Αφαίρεση του $f(x_0) = 3$ από κάθε αριθμό

Αφαιρούμε 3 από κάθε αριθμό:

$$(15-3, 10-3, 7-3, 8-3, 12-3, 21-3, 4-3, 6-3, 8-3, 3-3, \dots)$$

Σημείωση: Το 0 δεν αντιστοιχεί σε γράμμα, πιθανό σφάλμα στην κρυπτογράφηση.

Βήμα 3: Αντιστοίχιση αριθμών σε γράμματα

Χρησιμοποιούμε τον πίνακα αντιστοίχισης:

$12 \to$ μ
$7 \to$ η
$4 \to$ δ
$5 \to$ ε
$9 \to$ ι
$18 \to$ σ
$1 \to$ α
$3 \to$ γ
$5 \to$ ε
(0 δεν υπάρχει, πιθανό λάθος κρυπτογράφησης και θα έπρεπε να ειναι ω)
$12 \to$ μ
$5 \to$ ε
$19 \to$ τ
$17 \to$ ρ
$7 \to$ η
$19 \to$ τ
$15 \to$ ο
$18 \to$ σ
$5 \to$ ε
$9 \to$ ι
$18 \to$ σ
$9 \to$ ι
$19 \to$ τ
(0 δεν υπάρχει, πιθανό λάθος κρυπτογράφησης και θα έπρεπε να ειναι ω)

Άρα το αποκρυπτογραφημένο μήνυμα είναι: ¨μηδεισαγεωμετρητοςεισιτω'

Προφανώς, το σωστό μήνυμα είναι:¨**Μηδείς αγεωμέτρητος εισίτω'** , μια φράση που ήταν γραμμένη στην είσοδο της Ακαδημίας του Πλάτωνα.

Συμπέρασμα

Το σφάλμα στα ¨0' μπορεί να οφείλεται σε λάθος αρχική κρυπτογράφηση ή απλά σε μια μετατόπιση που δεν διατηρεί το αλφάβητο μέσα στα έγκυρα όρια (1-24). Ωστόσο, η λογική της αποκρυπτογράφησης ήταν σωστή και το τελικό μήνυμα επιβεβαιώνει την επιτυχή ανάκτηση του αρχικού κειμένου.

Άσκηση 2 (2.3)

Αυτή η μέθοδος χρησιμοποιεί:

1. Δοκιμή *Friedman*

Υπολογίζει τον **δείκτη σύμπτωσης (Ι῾)** για διαφορετικά μήκη κλειδιού ρ:

$$IC = \frac{\sum_{i=1}^{n} m_i(m_i - 1)}{k(k-1)}$$

όπου m_i είναι οι εμφανίσεις του i-οστού γράμματος και k το μήκος κειμένου. Όταν ο Ι῾ πλησιάζει την τιμή **0.0665** (τυπική για αγγλικά κείμενα), έχουμε πιθανό μήκος κλειδιού.

2. Ανάλυση Συχνότητας

Για κάθε ομάδα χαρακτήρων:

- Μετράμε τις συχνότητες εμφάνισης
- Δοκιμάζουμε όλες τις πιθανές μετατοπίσεις (0-25)
- Επιλέγουμε τη μετατόπιση που ελαχιστοποιεί την απόσταση από τις γνωστές αγγλικές συχνότητες

3. Αποκρυπτογράφηση

Εφαρμόζουμε το κλειδί $K = (k_1, k_2, ..., k_\rho)$ για να αναιρέσουμε τη μετατόπιση:

$$P_i = (C_i - k_{i \mod \rho}) \mod 26$$

Άσκηση 3 (2.4)

Λογική Επίλυσης

Έστω ότι δίνεται η σχέση κωδικοποίησης:

$$c = m \oplus (m \lll 6) \oplus (m \lll 10)$$

όπου $\lll$ δηλώνει κυκλική μετατόπιση προς τα αριστερά και $\oplus$ η πράξη XOR

Θέλουμε να βρούμε τον τύπο για την αποκωδικοποίηση, δηλαδή να εκφράσουμε το m ως συνάρτηση του c. Λόγω των ιδιοτήτων της XOR και της γραμμικότητάς της στο δυαδικό πεδίο ($GF(2)$), μπορούμε να επαναδιατυπώσουμε τη σχέση ως:

$$m = c \oplus (m \lll 6) \oplus (m \lll 10)$$

Αυτή η εξίσωση επιλύεται επαναληπτικά, ξεκινώντας με αρχική τιμή $m = 0$, και εφαρμόζοντας τον παραπάνω τύπο έως ότου το αποτέλεσμα σταθεροποιηθεί.

Περιγραφή Κώδικα *Python*

Ο κώδικας αποτελείται από τις παρακάτω βασικές συνιστώσες:

- $cyclic\_left\_shift(x, shift, bits)$: Υλοποιεί κυκλική μετατόπιση *αριστερά* $\ll$, για αριθμούς 16-βιτ.
- $encode(m)$: Υπολογίζει το c με βάση τον τύπο κωδικοποίησης.
- $decode(c)$: Ξεκινά με $m = 0$ και επαναλαμβάνει τον τύπο αποκωδικοποίησης για 16 επαναλήψεις, έως ότου υπολογιστεί το αρχικό μήνυμα m.
- **Επαλήθευση**: Παράγεται τυχαίο μήνυμα m, κωδικοποιείται σε c, και η αποκωδικοποίηση επαληθεύει την ορθότητα της μεθόδου με σύγκριση των τιμών m και m'.

Η μέθοδος είναι αποτελεσματική και συγκλίνει λόγω της γραμμικής φύσης της εξίσωσης και του περιορισμένου εύρους (16-βιτ) των αριθμών.

Άσκηση 4 (2.6)

Περιγραφή Κώδικα *OneTimePad*

Ο παρών αλγόριθμος υλοποιεί την κρυπτογράφηση και αποκρυπτογράφηση κειμένου με τη μέθοδο $OneTimePad(OTP)$ χρησιμοποιώντας προκαθορισμένη 5-βιτ δυαδική αναπαράσταση χαρακτήρων.

- Ορίζεται ένας πίνακας αντιστοίχισης χαρακτήρων σε $5-bit$ δυαδικά $strings$ και αντίστροφα.
- Το μήνυμα μετατρέπεται σε μία ακολουθία $bits$ με βάση τον πίνακα κωδικοποίησης.
- Δημιουργείται ένα τυχαίο δυαδικό κλειδί ίδιου μήκους με το μήνυμα.
- Η κρυπτογράφηση υλοποιείται με $bitwise$ τελεστή XOR ανάμεσα στο μήνυμα και το κλειδί.
- Το αποτέλεσμα της κρυπτογράφησης μετατρέπεται ξανά σε χαρακτήρες.
- Η αποκρυπτογράφηση υλοποιείται εφαρμόζοντας το XOR μεταξύ του κρυπτογραφημένου μηνύματος και του ίδιου κλειδιού, επιστρέφοντας το αρχικό μήνυμα.

Ο κώδικας περιλαμβάνει συναρτήσεις για:

1. Μετατροπή χαρακτήρων σε βιτς και αντίστροφα.
2. Δημιουργία τυχαίου κλειδιού.
3. Κρυπτογράφηση και αποκρυπτογράφηση με χρήση XOR.

Ο χρήστης εισάγει ένα μήνυμα και λαμβάνει ως έξοδο το κρυπτογραφημένο κείμενο, το κλειδί, καθώς και το αποκρυπτογραφημένο μήνυμα για επιβεβαίωση της ορθότητας.

Άσκηση 5 (3.3)

Χρησιμοποιήσαμε τη γεννήτρια $LCG(LinearCongruentialGenerator)$ με τις παρακάτω παραμέτρους:

- $m = 2^{10} = 1024$ (τροποποιητής)
- $a = 5$ (πολλαπλασιαστής)
- $c = 3$ (σταθερά)
- $x_0 = 1$ (αρχική τιμή)

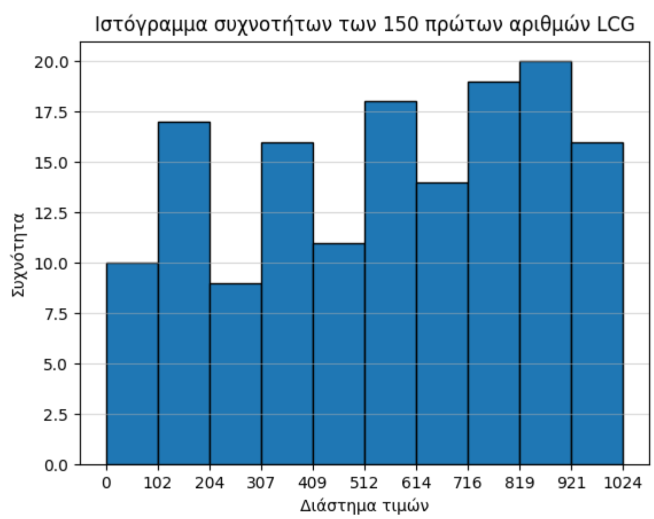
Η αναδρομική συνάρτηση που παράγει την ακολουθία είναι:

$$x_{i+1} = (a \cdot x_i + c) \mod m$$

Υπολογίστηκαν οι πρώτοι 150 όροι της ακολουθίας και αναλύθηκαν οι συχνότητες εμφάνισης των τιμών.

Ιστόγραμμα Συχνοτήτων

Το διάστημα τιμών $[0, 1023]$ χωρίστηκε σε 10 ισομεγέθη διαστήματα ($bins$), και για κάθε ένα μετρήθηκε πόσες φορές εμφανίστηκαν οι τιμές της ακολουθίας εντός αυτού. Το παρακάτω ιστόγραμμα δείχνει τις συχνότητες εμφάνισης:



Παρατηρήσεις και Συμπεράσματα

Η κατανομή των τιμών **δεν είναι απολύτως ομοιόμορφη**. Παρατηρήθηκαν διακυμάνσεις στις συχνότητες μεταξύ των διαστημάτων, ενώ ο αριθμός των διαφορετικών τιμών της ακολουθίας ήταν περιορισμένος σε σχέση με το πλήθος των δυνατών τιμών (1024). Αυτό οφείλεται στο ότι οι παράμετροι της LCG δεν εξασφαλίζουν μέγιστο μήκος κύκλου.

- Η ακολουθία δεν κάλυψε όλες τις τιμές στο διάστημα $[0, 1023]$.
- Ορισμένα διαστήματα είχαν εμφανώς περισσότερες τιμές.
- Η κατανομή αποκλίνει από την ιδανική ομοιόμορφη κατανομή.

Για να προσεγγιστεί καλύτερα η ομοιόμορφη κατανομή, προτείνεται:

- Χρήση παραμέτρων που εξασφαλίζουν **μέγιστο κύκλο** (π.χ. κατάλληλο a, c).
- Αύξηση του αριθμού παραγόμενων τιμών (π.χ. 1000 ή περισσότερες).

Άσκηση 6 (3.8)

- list_to_string(l): Ενώνει τα στοιχεία μιας λίστας σε ένα ενιαίο αλφαριθμητικό.
- string_xor(btext, key): Εκτελεί πράξη XOR μεταξύ δύο δυαδικών αλφαριθμητικών ίδιου μήκους.
- text_enc(text): Κωδικοποιεί ένα κείμενο σε δυαδική μορφή $5-bit$ ανά χαρακτήρα, σύμφωνα με το λεξικό aDict.
- text_dec(binary_string): Αποκωδικοποιεί μια δυαδική συμβολοσειρά σε κανονικό κείμενο, αντιστρέφοντας τη χαρτογράφηση του aDict.
- sumxor(l): Υπολογίζει το XOR όλων των στοιχείων μιας λίστας δυαδικών τιμών.
- lfsr(seed, feedback, bits): Υλοποιεί καταχωρητή ολίσθησης με γραμμική ανάδραση ($LFSR$), με αρχικό seed, θέσεις ανάδρασης feedback και πλήθος παραγόμενων $bits$
- find_seed(plaintext, ciphertext, feedback, aDict): Προσπαθεί να βρει τον αρχικό seed του $LFSR$, συγκρίνοντας το αναμενόμενο keystream που προκύπτει από το XOR του plaintext και ciphertext, με παραγόμενα keystreams.
- main(): Κεντρική $main$ που εκτελεί την κωδικοποίηση του μηνύματος, βρίσκει τον σωστό seed, παράγει το keystream και αποκωδικοποιεί το κρυπτογραφημένο μήνυμα.

Άσκηση 7 (3.11)

Περιγραφή Κώδικα

Ο κώδικας υλοποιεί την κρυπτογράφηση και αποκρυπτογράφηση ενός μηνύματος χρησιμοποιώντας τον αλγόριθμο $RC4$ και μία $5 - bit$ προσαρμοσμένη δυαδική αναπαράσταση χαρακτήρων.

1. Μετατροπές Χαρακτήρων

- $aDict$: Λεξικό που αντιστοιχεί χαρακτήρες σε 5-bit δυαδικό
- $reverseDict$: Αντιστροφή του $aDict$, μετατρέπει 5-βιτ δυαδικά σε χαρακτήρες.

2. $RC4$ $Key\ Scheduling\ Algorithm$ (KSA)

Η συνάρτηση $ksa(key)$ αρχικοποιεί τον πίνακα $S[0..255]$ με βάση το κλειδί $(HOUSE)$ και τον ανακατεύει χρησιμοποιώντας αλγοριθμική συνάρτηση.

3. $RC4$ $Pseudo - Random\ Generation\ Algorithm$ $(PRGA)$

Η συνάρτηση $prga(S, n)$ παράγει μια ακολουθία ν ψευδοτυχαίων $bytes$ $(keystream)$, που χρησιμοποιείται για το XOR με το μήνυμα.

4. Κωδικοποίηση και Αποκωδικοποίηση

- $text\_to\_binary(text)$: Μετατρέπει το μήνυμα σε συνεχόμενο δυαδικό $string$ χρησιμοποιώντας το $aDict$.
- $keystream\_to\_binary(keystream, length)$: Μετατρέπει κάθε $byte$ του $keystream$ σε 8-bit δυαδικό $string$ και το περικόπτει στο κατάλληλο μήκος.
- $xor(bin\_str, keystream)$: Υπολογίζει $bitwise$ XOR ανάμεσα στο δυαδικό μήνυμα και το $keystream$.
- $binary\_to\_text(binary)$: Μετατρέπει το αποκρυπτογραφημένο δυαδικό πίσω σε χαρακτήρες μέσω του $reverseDict$.

5. Ροή Προγράμματος

1. Το αρχικό κείμενο μετατρέπεται σε δυαδικό στρινγ 5-bit ανά χαρακτήρα.
2. Δημιουργείται το $keystream$ από το κλειδί με χρήση του $RC4$.
3. Πραγματοποιείται XOR για την κρυπτογράφηση.
4. Επαναλαμβάνεται το XOR για αποκρυπτογράφηση.
5. Το αποτέλεσμα μετατρέπεται πίσω σε αναγνώσιμο κείμενο.

Άσκηση 8 (4.3)

Η διαφορική ομοιομορφία ενός $S - box$, δηλαδή το $Diff(S)$, είναι ένα από τα πιο σημαντικά χαρακτηριστικά αντοχής του σε επιθέσεις διαφορικής κρυπτανάλυσης. Ορίζεται ως:

$$Diff(S) = \max_{\substack{x \in \{0,1\}^n \setminus \{0\} \\ y \in \{0,1\}^m}} |\{z \in \{0,1\}^n : S(z \oplus x) \oplus S(z) = y\}|$$

Δηλαδή, πρόκειται για το μέγιστο πλήθος λύσεων z για τις οποίες η διαφορική εξίσωση $S(z \oplus x) \oplus S(z) = y$ ισχύει, για κάθε μη μηδενικό x και για κάθε y.

Μια γενική κατώτερη εκτίμηση για τη διαφορική ομοιομορφία είναι:

$$Diff(S) \geq \max\left\{2, 2^{n-m}\right\}$$

Για ένα *S-box* που μετασχηματίζει από $n = 6$ *bit* σε $m = 4$ *bit*, δηλαδή $S : \{0,1\}^6 \to \{0,1\}^4$, έχουμε:

$$Diff(S) \geq \max\left\{2, 2^{6-4}\right\} = \max\{2, 4\} = 4$$

Αυτό σημαίνει πως οποιοδήποτε *S-box* που υλοποιεί τέτοιο μετασχηματισμό, θεωρητικά δεν μπορεί να έχει διαφορική ομοιομορφία μικρότερη του 4. Όσο μικρότερη είναι η τιμή του $Diff(S)$, τόσο πιο ανθεκτικό είναι το *S-box* στις διαφορικές επιθέσεις.

Σύμφωνα με την εκτέλεση του *script* που υλοποιεί τον παραπάνω υπολογισμό για το συγκεκριμένο *S-box* που δίνεται, βρέθηκε ότι:

$$Diff(S) = 14$$

Αυτό σημαίνει ότι για ορισμένες τιμές διαφορών εισόδου x και διαφορών εξόδου y, υπάρχουν έως και 14 διαφορετικές εισόδους z για τις οποίες η διαφορική εξίσωση ισχύει. Η τιμή αυτή είναι σημαντικά μεγαλύτερη από το ελάχιστο θεωρητικά επιτρεπτό όριο (4), και συνεπώς το συγκεκριμένο *S-box* είναι ευάλωτο σε διαφορικές επιθέσεις και δεν ενδείκνυται για χρήση σε σύγχρονα κρυπτοσυστήματα που απαιτούν ισχυρή αντίσταση.

Ο πίνακας κατανομής που προκύπτει από το *script* δείχνει την πλήρη κατανομή του πλήθους λύσεων της εξίσωσης για κάθε (x, y) και επιβεβαιώνει τη μέγιστη τιμή 14.

Άσκηση 9 (4.4)

Υπολογισμός του *Linear Branch Number* (*LBN*)

Ορίζουμε το $LinearBranchNumber(LBN)$ ενός S-κιβωτίου $S : \{0,1\}^m \to \{0,1\}^m$ ως:

$$LBN(S) = \min_{\alpha,\beta \in \mathbb{F}_2^m \setminus \{0\},\, C_S(\alpha,\beta) \neq 0} (wt(\alpha) + wt(\beta))$$

όπου:

- $wt(\cdot)$ είναι το βάρος $Hamming$ (πόσα $bits$ είναι 1),
- $C_S(\alpha,\beta) = \sum_{x \in \mathbb{F}_2^m} (-1)^{\beta \cdot S(x) + \alpha \cdot x}$ είναι ο συντελεστής συσχέτισης.

Ο στόχος είναι να υπολογίσουμε το LBN του S-κιβωτίου που ορίζεται από τον τύπο:

$$S(x) = (x^2 + 3) \mod 32$$

όπου x είναι η δεκαδική αναπαράσταση του δυαδικού x.

1. Ορισμός του LBN

Το LBN υπολογίζεται ως το ελάχιστο άθροισμα των βαρών $Hamming$ των διανυσμάτων α και β που οδηγούν σε μη μηδενική τιμή για τον συντελεστή συσχέτισης:

$$LBN(S) = \min_{\alpha,\beta \in \mathbb{F}_2^m \setminus \{0\},\, C_S(\alpha,\beta) \neq 0} (wt(\alpha) + wt(\beta))$$

Για να έχει $LBN(S) = 2$, πρέπει να βρούμε ζεύγη (α, β) που να ικανοποιούν:

$$wt(\alpha) + wt(\beta) = 2$$

και για τα οποία $C_S(\alpha,\beta) \neq 0$.

2. Πόσα τέτοια ζεύγη υπάρχουν·

Για κάθε διανύσμα $\alpha \in \mathbb{F}_2^5$ με βάρος 1, έχουμε:

$$\binom{5}{1} = 5 \quad \text{επιλογές για}\, \alpha$$

Το ίδιο ισχύει και για β, οπότε υπάρχουν συνολικά:

$$5 \times 5 = 25 \quad \text{πιθανοί συνδυασμοί}\, (\alpha, \beta)$$

με βάρος 2, δηλαδή:

$$wt(\alpha) + wt(\beta) = 2$$

3. Εξέταση της τιμής του $C_S(\alpha,\beta)$

Για κάθε ζεύγος (α, β), υπολογίζουμε τον συντελεστή συσχέτισης:

$$C_S(\alpha, \beta) = \sum_{x=0}^{31} (-1)^{\beta \cdot S(x) + \alpha \cdot x}$$

Από τα 25 ζεύγη, μόνο **8** έχουν μη μηδενική τιμή για τον συντελεστή συσχέτισης $C_S(\alpha, \beta) \neq 0$.

4. Συμπέρασμα

Η ελάχιστη τιμή που μπορεί να έχει το LBN είναι 2, και αυτή επιτυγχάνεται μόνο για 8 ζεύγη (α, β). Επομένως, το $LinearBranchNumber$ του S-κιβωτίου είναι:

$$LBN(S) = 2$$

Αυτό δείχνει ότι το S-κιβώτιο προσφέρει μια μέτρια σύγχυση ($confusion$), καθώς το LBN δεν είναι κοντά στην μέγιστη τιμή που είναι 6 (δηλαδή $m + 1$).

Άσκηση 10 (4.8)

0.1 Ανάλυση του $Avalanche\ Effect$ στον $AES - 128$

Το $avalanche\ effect$ είναι μία βασική ιδιότητα που πρέπει να πληρεί κάθε ασφαλής αλγόριθμος κρυπτογράφησης. Στην περίπτωση του $AES - 128$, το $avalanche\ effect$ αναφέρεται στη συμπεριφορά του αλγορίθμου όταν το αρχικό μήνυμα (ή $plaintext$) υποβάλλεται σε μία μικρή τροποποίηση (π.χ., η αλλαγή ενός μόνο bit). Η ιδέα είναι ότι μία τόσο μικρή αλλαγή στο αρχικό μήνυμα θα πρέπει να προκαλέσει σημαντική αλλαγή στο κρυπτογραφημένο κείμενο ($ciphertext$), με το αποτέλεσμα να είναι ότι σχεδόν όλα τα $bits$ του κρυπτογραφημένου κειμένου να αλλάξουν.

0.2 Δημιουργία Ζευγαριών Μηνυμάτων

Για την ανάλυση του $avalanche\ effect$, δημιουργήσαμε πάνω από 30 ζευγάρια μηνυμάτων m_1 και m_2, όπου τα μηνύματα διαφέρουν σε ακριβώς 1 bit. Συγκεκριμένα:

- Τα μηνύματα m_1 και m_2 έχουν μήκος 256 $bits$ (32 $bytes$), το οποίο είναι διπλάσιο από το μήκος ενός $block$ του $AES - 128$ (128 $bits$ ή 16 $bytes$).
- Το μήνυμα m_2 προκύπτει από το μήνυμα m_1 αλλάζοντας μόνο ένα bit.

0.3 Κρυπτογράφηση με $AES - 128$

Η κρυπτογράφηση των μηνυμάτων έγινε χρησιμοποιώντας τον αλγόριθμο $AES - 128$ και δύο καταστάσεις λειτουργίας:

- ECB ($Electronic\ Codebook$): Κρυπτογράφηση με ανεξάρτητη κρυπτογράφηση κάθε μπλοκ.

- *CBC* (*Cipher Block Chaining*): Κρυπτογράφηση με αλυσίδωση των μπλοκ και χρήση του *IV* (*Initialization Vector*).

Για κάθε ζευγάρι μηνυμάτων (m_1, m_2), κρυπτογραφήθηκαν τα μηνύματα και στη συνέχεια υπολογίστηκε η διαφορά σε *bits* μεταξύ των αντίστοιχων κρυπτογραφημένων μηνυμάτων.

0.4 Υπολογισμός Διαφοράς σε *Bits*

Η διαφορά σε *bits* μεταξύ δύο κρυπτογραφημένων μηνυμάτων υπολογίστηκε χρησιμοποιώντας την *Hamming distance*, η οποία μετρά πόσα *bits* διαφέρουν μεταξύ δύο κωδικοποιημένων δεδομένων. Στην περίπτωση αυτή, η διαφορά σε *bits* για κάθε ζευγάρι μηνυμάτων υπολογίστηκε για τις λειτουργίες *ECB* και *CBC*.

0.5 Αποτελέσματα και Ανάλυση

Τα αποτελέσματα του τεστ έδειξαν ότι:

- Στην *ECB*, αν και τα μπλοκ κρυπτογραφούνται ανεξάρτητα το ένα από το άλλο, η διαφορά στα κρυπτογραφημένα μηνύματα για κάθε ζευγάρι μηνυμάτων ήταν σχετικά υψηλή, αν και ίσως όχι τόσο έντονη όσο στην *CBC*.
- Στην *CBC*, η διαφορά στα κρυπτογραφημένα μηνύματα ήταν πιο έντονη λόγω της αλυσίδωσης των μπλοκ και της χρήσης του Ι΅. Η αλλαγή ενός *bit* στο αρχικό μήνυμα προκαλεί μεγάλες αλλαγές όχι μόνο στο αντίστοιχο μπλοκ, αλλά και στο επόμενο μπλοκ.

Η μέση διαφορά σε *bits* που παρατηρήθηκε ήταν κοντά στο 50% του μήκους του κρυπτογραφημένου μηνύματος (128 *bits*), το οποίο επιβεβαιώνει την ύπαρξη του αaλανςηε εφφεςτ στον ΑΕΣ-128.

0.6 Συμπέρασμα

Από τα αποτελέσματα της ανάλυσης, μπορούμε να συμπεράνουμε ότι ο *AES*-128 εκπληρώνει την απαίτηση για το *avalanche effect*, με την αλλαγή ενός μόνο *bit* στο αρχικό μήνυμα να προκαλεί σημαντική αλλαγή στο κρυπτογραφημένο κείμενο. Η λειτουργία *CBC* προσφέρει ισχυρότερο *avalanche effect* σε σχέση με την *ECB*, καθώς η αλυσίδωση των μπλοκ και η χρήση του *IV* ενισχύουν την ασφάλεια του αλγορίθμου.

0.7 Ανάλυση Κώδικα

- **Κύριες Συναρτήσεις**:
  - *hamming_distance*(*a*, *b*): Υπολογίζει την απόσταση *Hamming* μεταξύ δύο *bytestrings*.
  - *main_execution_block*: Δημιουργεί μηνύματα, εφαρμόζει $bit - flip$, και εκτελεί κρυπτογραφήσεις.

- **Βασικές Μεταβλητές**:
  - *key*: 128-*bit* τυχαίο κλειδί (*AES*-128)
  - *iv*: 128-*bit* τυχαίο διάνυσμα αρχικοποίησης (*CBC*)
  - *msg_length*: 32 *bytes* (256 *bits*, 2 *blocks*)

0.8 Γενική Λειτουργία

1. Δημιουργία 30 ζευγαριών μηνυμάτων (m_1, m_2) με διαφορά 1 *bit*.
2. Κρυπτογράφηση σε δύο λειτουργίες:
   - *ECB*: Κάθε *block* κρυπτογραφείται ανεξάρτητα.
   - *CBC*: Χρήση *IV* και αλυσιδωτής σύνδεσης *blocks*.
3. Υπολογισμός διαφοράς *bits* στα κρυπτογραφήματα.

Σημείωση

Για την ευκολία της συγγραφής και την αισθητική παρουσίαση της αναφοράς, χρησιμοποιήθηκε σε μικρό βαθμό η βοήθεια του *ChatGPT*, κυρίως:

- για τη μορφοποίηση του αρχείου LaTeX και το στήσιμο του ιστογράμματος,
- για τη βελτίωση κάποιων κομματιών κώδικα, όπως οι περιττοί και χρονοβόροι έλεγχοι τα *labels* κλπ.
- για εύρεση επιπρόσθετων πληροφοριών

Η κατανόηση και η επίλυση των ασκήσεων έγιναν από εμένα και η χρήση του εργαλείου έγινε απλώς για διευκόλυνση και όχι για αντικατάσταση της διαδικασίας μάθησης.