Παραθυρική Εφαρμογή JAVA του Αλγορίθμου Κρυπτογράφησης Ροής RC4

Η υλοποίηση έγινε στα πλαίσια του ΠΜΣ Πληροφορικής

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Πρόγραμμα Μεταπτυχιακών Σπουδών στην Πληροφορική
Επικοινωνιακά Συστήματα και Τεχνολογίες
Προχωρημένα Θέματα Ασφάλειας

Δημήτριος Ρούσης (ΑΜ: 532)

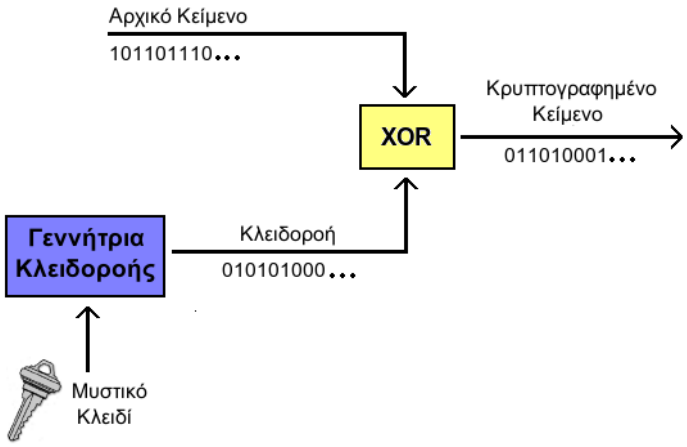
ΠΕΡΙΛΗΨΗ

Στην παρούσα προγραμματιστική εργασία υλοποιείται ο αλγόριθμος κρυπτογράφησης ροής, RC4. Πρόκειται για εφαρμογή με γραφικό περιβάλλον, γραμμένη στην δημοφιλή γλώσσα αντικειμενοστραφούς προγραμματισμού, JAVA. Η κρυπτογράφηση και η αποκρυπτογράφηση γίνονται με ευκολία και ταχύτητα χρησιμοποιώντας απλά το ποντίκι του υπολογιστή. Παρέχεται ολόκληρος ο πηγαίος κώδικας καθώς και εκτελέσιμα αρχεία για διαφορετικές αρχιτεκτονικές επεξεργαστών. Τέλος υπάρχει ιστοσελίδα υποστήριξης της εφαρμογής για νέες εκδόσεις και επίλυση προβλημάτων των προγραμματιστών και των φοιτητών που ασχολούνται με το αντίστοιχο αντικείμενο..

ΘΕΩΡΗΤΙΚΟ ΜΕΡΟΣ

1. Εισαγωγή

Η δομή ενός αλγόριθμου κρυπτογράφησης ροής συνοψίζεται στην εικόνα που ακολουθεί:



Τα κυριότερα χαρακτηριστικά των αλγορίθμων κρυπτογράφησης ροής είναι τα παρακάτω:

- Επεξεργάζονται το μήνυμα byte προς byte. Αν και αυτό είναι επίσης εφικτό σε οποιοδήποτε μέγεθος δομής (bit προς bit, ως μια ροή)
- Παρουσιάζουν όμοια λειτουργία με αυτή των αλγορίθμων one-time pad, με τη διαφορά ότι έχουν ένα ψευδοτυχαίο keystream k που παράγεται από αλγόριθμο PRNG (Pseudorandom Number Generator) με βάση ένα κλειδί K
- Για την δημιουργία του κρυπτογραφημένου κειμένου C, το keystream k γίνεται XOR με το plaintext M bit προς bit: $C_i = M_i$ XOR k_i
- Ενώ για αποκρυπτογράφηση γίνεται η αντίστροφη διαδικασία: $M_i = C_i$ XOR k_i
- Η τυχαιότητα του keystream καταστρέφει τελείως τις στατιστικές ιδιότητες του μηνύματος
- Δεν πρέπει ποτέ να επαναχρησιμοποιούμε το ίδιο keystream

Η έξοδος του PRNG (keystream) θα πρέπει να έχει τις ακόλουθες ιδιότητες:

- Μακρά περίοδο επανάληψης. Όσο μεγαλύτερη είναι αυτή η περίοδος τόσο δυσκολότερη θα είναι και η κρυπτανάλυση. Επειδή ουσιαστικά η γεννήτρια κλειδοροής είναι μία μαθηματική συνάρτηση που δέχεται ως είσοδο το μυστικό κλειδί και παράγει στην έξοδο την κλειδοροή, είναι βέβαιο πως η κλειδοροή που θα παραχθεί θα είναι από ένα σημείο και μετά περιοδική. Αυτό σημαίνει πως μετά από κάποιον αριθμό bits της κλειδοροής, αυτή θα επαναλαμβάνεται ξεκινώντας από την αρχή.
- Να είναι όσο το δυνατό πιο όμοια σε πραγματικό RNG stream. Αυτό σημαίνει ότι η μαθηματική συνάρτηση που χρησιμοποιείται στην γεννήτρια κλειδοροής θα πρέπει να επιλεγεί κατάλληλα ούτως ώστε το αποτέλεσμά της να πλησιάζει όσο το δυνατόν περισσότερο το τυχαίο. Υπάρχουν ειδικές μέθοδοι δοκιμής της καταλληλότητας της γεννήτριας κλειδοροής, οι οποίοι εκπονούν ελέγχους τυχαιότητας (randomness tests) σε αυτήν. Ένας έλεγχος τυχαιότητας είναι για παράδειγμα ο εξής: Για μία κλειδοροή μήκους εκατομμυρίων bits, θα πρέπει ο αριθμός των 1 να να είναι παρόμοιος με τον αριθμό των 0. Επίσης θα πρέπει κάθε 0 να ακολουθείται από 1 τόσο συχνά όσο και το αντίστροφο. Δηλαδή να υπάρχουν ανεξάρτητες εμφανίσεις 0 και 1 (ίδιο πλήθος διαφορετικών 8δων bits).
- Να γίνεται χρήση επαρκώς μεγάλου κλειδιού (>128 bit). Για την παραγωγή μεγάλου πλήθους διαφορετικών keystreams k. Αυτό το χαρακτηριστικό θα παρέχει προστασία από επιθέσεις brute-force.

Το βασικό πλεονέκτημα των αλγορίθμων κρυπτογράφησης ροής είναι ότι είναι απλούστεροι από άλλων ειδών κρυπταλγόριθμους με αποτέλεσμα η εκτέλεσή τους να είναι πολύ ταχύτερη. Για το λόγο αυτό χρησιμοποιούνται ευρέως σε ενσωματωμένα συστήματα (8 ή 32bit) αλλά και σε δημοφιλή πρωτόκολλα επικοινωνίας όπως είναι για παράδειγμα το WiFi.

2. Ο αλγόριθμος ροής RC4 (ή ARC4 ή ARCFOUR)

Ο αλγόριθμος κρυπτογράφησης ροής RC4 είναι ένας ιδιωτικός κρυπτογραφικός αλγόριθμος που ανήκει στην RSA Security. Πρόκειται για κρυπταλγόριθμο ροής, μεταβλητού μήκους κλειδιού και διεργασίες βασισμένες σε bytes.

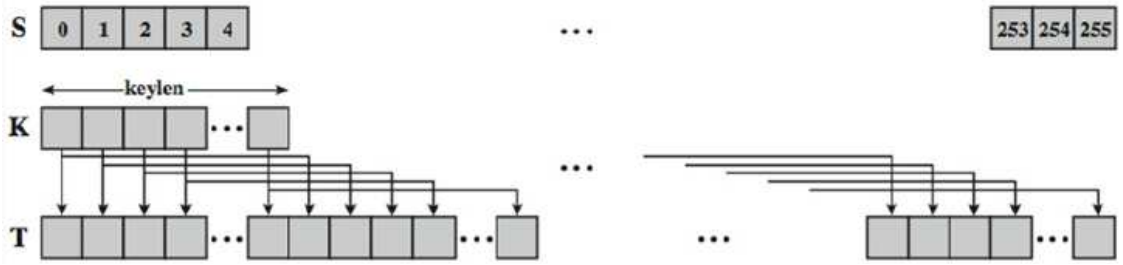
Η σχεδίαση του είναι απλή αλλά αποτελεσματική. Ουσιαστικά υλοποιεί έναν αλγόριθμο PRNG για κρυπτογράφηση ροής (stream ciphering) με τη μέθοδο του τυχαίου ανακατέματος (random permutation).

Λόγω τη απλότητας και της ταχύτητάς του, ο αλγόριθμος RC4 χρησιμοποιείται ευρύτατα σε γνωστά και δημοφιλή πρωτόκολλα στο web (πχ: SSL/TLS), αλλά και σε ασύρματα πρωτόκολλα επικοινωνίας όπως το WiFi για τα WEP και WPA.

Η λειτουργία του ξεκινά με ένα πίνακα S που περιέχει τους αριθμούς 0 έως 255, τοποθετημένους σε αύξουσα σειρά. Χρησιμοποιεί ένα κλειδί μεταβλητού μήκους (1-256 bytes) βάσει του οποίου αρχικοποιείται ένας προσωρινός πίνακας T, 256 θέσεων. Το κλειδί περιέχεται και αυτό σε έναν πίνακα K που το μέγεθός του εξαρτάται από το μήκος του κλειδιού αυτού. Κάθε θέση των πινάκων S, K και T είναι μεγέθους ενός byte.

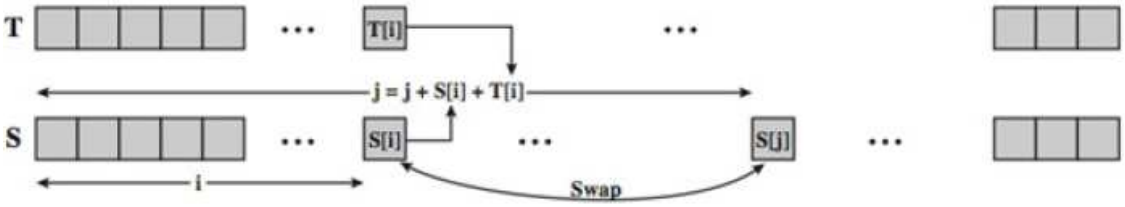
Η αρχικοποίηση αυτών των πινάκων περιγράφεται αλγοριθμικά και σχηματικά από τον παρακάτω ψευδοκώδικα και την εικόνα:

```
/* Initialization*/
for i = 0 to 255 do
S[i] = i
T[i] = K[i mod keylen];
```



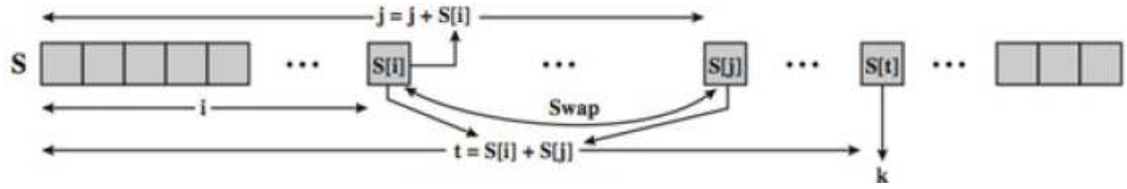
Στη συνέχεια γίνεται το λεγόμενο "αρχικό ανακάτεμα" του πίνακα S με τη βοήθεια του πίνακα T. Ο πίνακας S δημιουργεί την αρχική εσωτερική κατάσταση του αλγορίθμου. Πιθανοτικά είναι εύκολο να γίνει αντιληπτό, ότι οι πιθανές αρχικές καταστάσεις που μπορεί να λάβει ο S, ισούνται με 256! (δηλαδή 256 παραγοντικό). Η διαδικασία του αρχικού ανακατέματος φαίνεται σχηματικά και αλγοριθμικά παρακάτω:

```
/* Initial permutation of S */
j = 0
for i = 0 to 255 do
j = (j + S[i] + T[i]) (mod 256)
swap (S[i], S[j])
```



Για να κρυπτογραφήσουμε ένα byte plaintext, η διαδικασία συνεχίζεται αντιμεταθέτοντας για κάθε κρυπτογράφηση ένα ζεύγος τιμών του S. Το άθροισμα t των περιεχομένων του ζεύγους που αντιμετατίθεται ορίζει την τιμη του "stream key" S[t]. Τέλος, Κανουμε XOR το S[t] με το επόμενο byte του plaintext για να το κρυπτογραφήσουμε / αποκρυπτογραφήσουμε. Όλα αυτά φαίνονται καλύτερα πιο κάτω:

```
/* Keystream Generation */
i = j = 0
for each message byte Mi
i = (i + 1) (mod 256)
j = (j + S[i]) (mod 256)
swap(S[i], S[j])
t = (S[i] + S[j]) (mod 256)
Ci = Mi XOR S[t]
```



2.1 Η ασφάλεια του RC4

Ο RC4 είναι ασφαλής έναντι των γνωστών επιθέσεων (πχ: Brute Force Attack). Έχουν υπάρξει κάποιες προσπάθειες κρυπτανάλυσης, αλλά καμία δεν είναι πρακτική για λογικό μέγεθος κλειδιού (>=128 bits).

Ο RC4 είναι αλγόριθμος κρυπτογράφησης ροής κι έτσι κληρονομεί τα βασικά χαρακτηριστικά αυτής της οικογένειας αλγορίθμων. Έτσι για παράδειγμα, δε θα πρέπει ποτέ να χρησιμοποιεί το ίδιο κλειδί. Αυτό επιλύεται σχεδόν τέλεια με γεννήτριες τυχαίων κλειδιών (RNGs).

ΠΡΑΚΤΙΚΟ ΜΕΡΟΣ

1. Τεκμηρίωση

Ο πηγαίος κώδικας της εφαρμογής χωρίζεται σε δύο τμήματα:

1. Βασικές συναρτήσεις του κρυπταλγόριθμου RC4 (RC4alg.java)
2. Συναρτήσεις γραφικού περιβάλλοντος χρήστη (RC4.java)

Οι βασικές συναρτήσεις του RC4 φροντίζουν για την αρχικοποίηση και τη δημιουργία του κλειδιού, αλλά και για την διαδικασία κρυπτογράφησης και αποκρυπτογράφησης. Ενώ στον κώδικα του γραφικού περιβάλλοντος περιλαμβάνεται η κύρια συνάρτηση main(), οι συναρτήσεις δημιουργίας της παραθυρικής διεπαφής χρήστη (GUI API) και οι διαδικασίες κλήσης των συναρτήσεων κρυπτογράφησης και αποκρυπτογράφησης.

1.1 Βασικές συναρτήσεις του κρυπταλγόριθμου RC4

Στο αρχείο κλάσης RC4alg.java περιλαμβάνονται τέσσερις συναρτήσεις:

```
1. RC4alg(int key[])
2. int[] encrypt(int plaintext[])
3. int[] decrypt(int cipher[])
4. filepath()
```

Οι πρώτες 3 είναι οι κύριες και η τέταρτη είναι βοηθητική.

1.1.1 Συνάρτηση `RC4alg(int key[])`

Πηγαίος κώδικας:

```
/* Main Constructor - Init function */
public RC4alg(int key[])
{
    int j, swap;
```

```
        /* Initialization */
        for(int i=0; i<256; i++)
        {
           S[i]= i;
           T[i]= key[i % key.length];
        }

        /* Initial permutation of S */
        j = 0;
        for(int i=0; i<256; i++)
        {
           j=(j + S[i] + T[i]) % 256;
           /* classic swap */
           swap = S[i];
           S[i] = S[j];
           S[j] = swap;
        }
    }
```

Λειτουργία:

Η συνάρτηση αυτή επιτελεί κύρια λειτουργικότητα καθώς εκτός του ότι λειτουργεί ως constructor της κλάσης, είναι υπεύθυνη και για την αρχικοποίηση των πινάκων που χρησιμοποιεί ο RC4 αλλά και για το ανακάτεμα του πίνακα S. Ως είσοδο λαμβάνει το κλειδί κρυπτογράφησης του χρήστη.

1.1.2 Συνάρτηση `int[] encrypt(int plaintext[])`

Πηγαίος κώδικας:

```
    /* RC4 encrypt function */
    public int[] encrypt(int plaintext[])
    {
       int ciphertext[];
       ciphertext = new int[plaintext.length];
       int i, j, swap, t;

       i = j = 0 ;
       for(int count=0; count<plaintext.length; count++)
       {
          i = (i + 1) % 256;
          j = (j + S[i]) % 256;

          /* classic swap */
          swap = S[i];
          S[i] = S[j];
          S[j] = swap;

          t = (S[i] + S[j]) % 256;

          ciphertext[count] = plaintext[count] ^ S[t];
       }
       return ciphertext;
    }
```

Λειτουργία:

Η συνάρτηση δέχεται ως είσοδο το μη-κρυπτογραφημένο κείμενο (plaintext) που εισάγει ο χρήστης και ως έξοδο δίνει το αντίστοιχο κρυπτογραφημένο κείμενο (ciphertext).

1.1.3 Συνάρτηση `int[] decrypt(int cipher[])`

Πηγαίος κώδικας

```
/* Decrypt function */
public int[] decrypt(int cipher[])
{
    return encrypt(cipher);
}
```

Λειτουργία

Η συνάρτηση της αποκρυπτογράφησης ουσιαστικά επιτελεί κρυπτογράφηση του κρυπτογραφημένου κειμένου που λαμβάνει ως είσοδο και παράγει ως έξοδο το αντίστοιχο αρχικό, μη-κρυπτογραφημένο κείμενο. Η υλοποίηση αυτή βασίζεται στη θεωρία των αλγορίθμων κρυπτογράφησης ροής (Stream Ciphers) που αναφέρεται και στο θεωρητικό μέρος αυτού του εγγράφου.

1.1.4 Η Συνάρτηση `getFilepath()`

Πηγαίος κώδικας:

```
public static String getFilepath()
{
    /* Load file dialog box */
    Frame frame = null;
    FileDialog fDialog = new FileDialog(frame, "Open", FileDialog.LOAD);
     fDialog.setVisible(true);
     /* Calculate filename and directory of the file */
     String filename = fDialog.getFile();
     String directory = fDialog.getDirectory();
     /* Concatenate filename and directory into pathname */
     String path = directory + filename;
     /* Return the string of the pathname for further use */
    return path;
}
```

Λειτουργία:

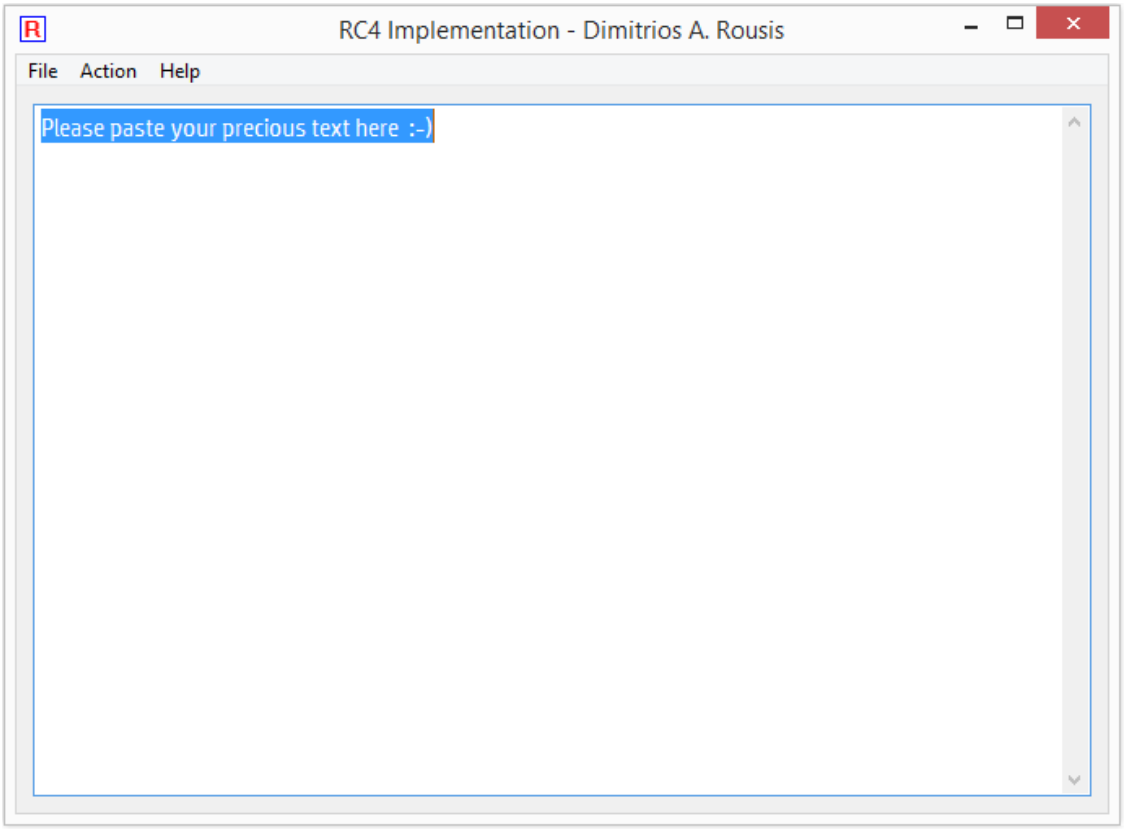
Όπως αναφέρθηκε, η συνάρτηση αυτή είναι βοηθητική και επομένως δε σχετίζεται άμεσα με τη λειτουργία του RC4. Πρόκειται ουσιαστικά για ένα σύνολο κλήσεων άλλων συναρτήσεων της JAVA προκειμένου ο χρήστης να μπορεί να υποδείξει στην εφαρμογή ποιο αρχείο θέλει να κρυπτογραφήσει. Ως είσοδο λαμβάνει ένα αρχείο και ως έξοδο επιστρέφει την απόλυτη διαδρομή (absolute file path) του αρχείου αυτού στο δίσκο.

1.1.5 Εσωτερική τεκμηρίωση

Με τον όρο εσωτερική τεκμηρίωση εννοούμε τα σχόλια που περιλαμβάνονται ενσωματωμένα στον πηγαίο κώδικα και τα οποία βοηθούν στην κατανόηση του τί γίνεται, πως γίνεται και γιατί γίνεται. Αυτό βοηθά τους υπόλοιπους προγραμματιστές αλλά και τον ίδιο το δημιουργό να μπορούν να βελτιώνουν, να επεκτείνουν και να συντηρούν την εφαρμογή ακόμη και μετά από την παρέλευση ετών.

2. Το γραφικό περιβάλλον

Το γραφικό περιβάλλον της εφαρμογής παρέχει ευκολία και ταχύτητα στον χρήστη. Όπως φαίνεται και στην εικόνα που ακολουθεί, αποτελείται από την κεντρική μπάρα επιλογών (main menu) και ένα πεδίο κειμένου (textarea) στο οποίο ο χρήστης μπορεί να εισάγει το κείμενό του:



Η κεντρική μπάρα περιλαμβάνει τρία μενού μέσω των οποίων ο χρήστης χειρίζεται εξ' ολοκλήρου την εφαρμογή. Τα μενού αυτά διαιρούνται σε επιμέρους υπομενού. Η δομή τους είναι η εξής:

- File
    - New
    - Open
    - Save
    - Exit

- Action
    - Encrypt
    - Decrypt
    - Brute Force Analysis > (δεν υποστηρίζεται σε αυτή την έκδοση)
- Help
    - Documentation
    - About

Σε επόμενη παράγραφο παρατίθεται ο πηγαίος κώδικας των υπομενού και εξηγείται η λειτουργικότητά τους.

2.1 Ο πηγαίος κώδικας του γραφικού περιβάλλοντος

Νωρίτερα παρατέθηκε ο πηγαίος κώδικας του αρχείου RC4alg.java στον οποίο περιλαμβάνονται όλες οι απαραίτητες συναρτήσεις για την επίτευξη των βασικών λειτουργιών κρυπτογράφησης και αποκρυπτογράφησης του κρυπταλγόριθμου ροής RC4.

Οι συναρτήσεις αυτές από μόνες τους δεν είναι δυνατόν να παρέχουν κάποια λειτουργικότητα. Όπως συμβαίνει στις περισσότερες γλώσσες προγραμματισμού, οι είσοδοί τους θα πρέπει να αρχικοποιηθούν κατάλληλα μέσω κώδικα που θα περιλαμβάνεται με την σειρά του σε μια κύρια (main) συνάρτηση. Στο αρχείο RC4.java συμβαίνει ακριβώς αυτό. Λόγω όμως της γραφικής προσέγγισης, στο ίδιο αρχείο περιγράφεται και το παραθυρικό περιβάλλον διεπαφής χρήστη.

Ο σκελετός της δομής του κώδικα είναι ο παρακάτω:

```
package rc4;
...
import java.awt.Desktop;
...
public class RC4 {
	...
	public static void main(String[] args) throws IOException {
	...
	}
	public void open() {
	...
	}
```

```
	protected void createContents()
	{
	}
}
```

Γίνεται αντιληπτό ότι η κεντρική κλάση (class) είναι η RC4 μέσα στην οποία υπάρχει εμφωλευμένος όλος ο κώδικας. Επίσης διακρίνονται τρείς μέθοδοι:

1. `public static void main(String[] args)`
2. `public void open()`
3. `protected void createContents()`

Η main() καλεί την open(), η οποία με τη σειρά της καλεί την createContents().

2.1.1 Η μέθοδος `main()`

Πρόκειται για την κύρια μέθοδο, μέσω της οποίας ξεκινούν και τερματίζονται τα πάντα. Ο πηγαίος κώδικας παρουσιάζει από μόνος του την λειτουργικότητά της:

```
public static void main(String[] args) throws IOException {
	try {
		RC4 window = new RC4();
		window.open();
	} catch (Exception e) {
		e.printStackTrace();
	}
}
```

Δημιουργεί ένα αντικείμενο window, δηλαδή το παράθυρο της εφαρμογής μας, το οποίο είναι τύπου RC4, δηλαδή έχει τα χαρακτηριστικά της κεντρικής κλάσης. Στη συνέχεια ανοίγει αυτό το παράθυρο μέσω της open().

Όλες οι υπόλοιπες μέθοδοι που θα αναλυθούν πιο κάτω, και πιο συγκεκριμένα το αποτέλεσμα της λειτουργίας τους, δημιουργούν τα χαρακτηριστικά και τη συμπεριφορά αυτού του παραθύρου.

2.1.2 Η μέθοδος open( )

Όπως προαναφέραμε στην προηγούμενη παράγραφο, η μέθοδος open() καλείται προκειμένου να ανοίξει το κεντρικό παράθυρο της εφαρμογής. Ο πηγαίος της κώδικας είναι:

```
public void open() {
	Display display = Display.getDefault();
	createContents();
	shlRcImplementation.open();
	shlRcImplementation.layout();
	while (!shlRcImplementation.isDisposed()) {
		if (!display.readAndDispatch()) {
			display.sleep();
		}
	}
}
```

Μέσω αυτής της μεθόδου, καλείται κατά κύριο λόγο η μέθοδος createContents().

2.1.3 Η μέθοδος createContents( )

Στη μέθοδο αυτή περιγράφονται σχεδόν όλα τα γραφικά χαρακτηριστικά του κεντρικού παραθύρου της εφαρμογής και η λειτουργικότητα των στοιχείων που υπάρχουν σε αυτό.

Λόγω του ότι η συγκεκριμένη εφαρμογή είναι πολύ μικρή, σχεδόν όλα γίνονται από τις επιλογές που διαθέτει το κεντρικό μενού. Ο κύριος σκελετός του κώδικα της μεθόδου createContents() είναι ο εξής:

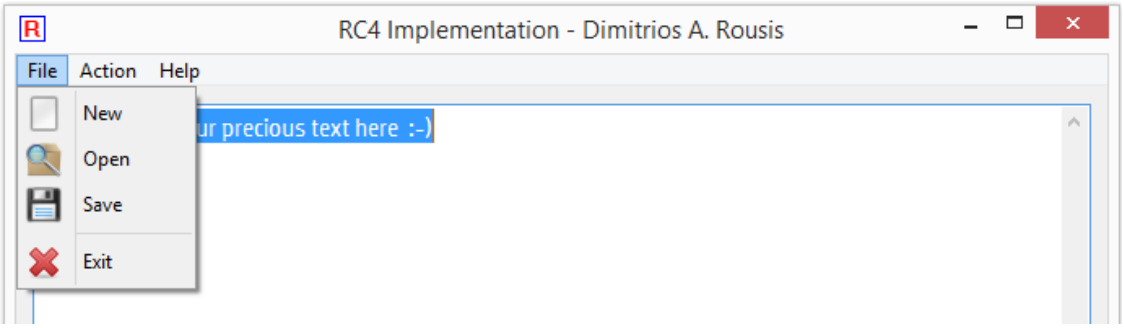
```
shlRcImplementation = new Shell();
...
/* Create textarea when the app starts */
text = new Text(shlRcImplementation, SWT.BORDER | SWT.WRAP | SWT.V_SCROLL |
SWT.MULTI);
...
/* Menu Bar - Contains all menus */
Menu menu = new Menu(shlRcImplementation, SWT.BAR);
...
/* Start of Cascade Menu "File" - Contains "New", "Open", "Save" and "Exit"
MenuItems */
MenuItem cascadeMenuFile = new MenuItem(menu, SWT.CASCADE);
...
/* MenuItem: New */
MenuItem New = new MenuItem(menuFile, SWT.NONE);
...
```

```
/* MenuItem: Open */
MenuItem Open = new MenuItem(menuFile, SWT.NONE);
...
/* MenuItem: Save */
MenuItem Save = new MenuItem(menuFile, SWT.NONE);
...
/* Separator */
new MenuItem(menuFile, SWT.SEPARATOR);
...
/* MenuItem: Exit */
MenuItem Exit = new MenuItem(menuFile, SWT.NONE);
/* End of Cascade Menu "File" */
...
/* Start of Cascade Menu "Action" - Contains "Encrypt", "Decrypt" and "Brute
Force Analysis" MenuItems */
MenuItem cascadeMenuAction = new MenuItem(menu, SWT.CASCADE);
...
/* MenuItem: Encrypt */
MenuItem Encrypt = new MenuItem(menuAction, SWT.NONE);
...
/* MenuItem: Decrypt */
MenuItem Decrypt = new MenuItem(menuAction, SWT.NONE);
...
/* MenuItem: Brute Force Analysis */
MenuItem BruteForceAnalysis = new MenuItem(menuAction, SWT.NONE);
...
/* End of Cascade Menu "Action" */
...
/* Start of Cascade Menu "Help" - Contains "Documentation", and "About"
MenuItems */
MenuItem cascadeMenuHelp = new MenuItem(menu, SWT.CASCADE);
...
/* MenuItem: Documentation */
MenuItem Documentation = new MenuItem(menuHelp, SWT.NONE);
...
/* MenuItem: About */
MenuItem About = new MenuItem(menuHelp, SWT.NONE);
...
/* End of Cascade Menu "Help" */
/* End of Menu Bar - Contains all menus */
```

2.2 Περιγραφή πηγαίου κώδικα των υπομενού

Σε αυτή την παράγραφο περιγράφεται ο πηγαίος κώδικας των υπομενού. Τα σημαντικότερα εξ' αυτών είναι εκείνα που ανήκουν στο μενού Action καθώς επιτελούν και τις διεργασίες για τις οποίες γράφτηκε ολόκληρη η εφαρμογή.

Θα ξεκινήσουμε από το μενού File. Όπως φαίνεται και στην εικόνα, σε αυτό υπάρχουν τρεις βασικές λειτουργίες χειρισμού αρχείων και μία βοηθητική για τον τερματισμό της εφαρμογής:



2.2.1 Υπομενού New

Επιλέγοντας New, η περιοχή κειμένου καθαρίζει από ότι είχε προηγουμένως προκειμένου να φιλοξενήσει το καινούριο περιεχόμενο που ο χρήστης επιθυμεί να προσθέσει. Ο πηγαίος κώδικας του υπομενού αυτού είναι:

```
text.setText("Please paste your precious text here  :-)");
text.selectAll();
```

2.2.2 Υπομενού Open

Επιλέγοντας Open, εμφανίζεται παράθυρο διαλόγου μέσω του οποίου ο χρήστης μπορεί να ανοίξει κάποιο αρχείο κειμένου από κάποιο αποθηκευτικό μέσο. Ο πηγαίος κώδικας του υπομενού αυτού είναι:

```
/* 1. Find out the pathname of */
String filepath = RC4alg.getFilepath();
/* 2. Clear the contents of the textaarea */
text.setText("");
/* 3. Read the contents of the text file opened */
BufferedReader br = new BufferedReader(new FileReader(filepath));
String text1 = "";
String line = br.readLine();
/* 4. Print the contents of the text file to the text area */
int len = line.length();
```

```
for(int i=0; i<len; i++)
{
	text.append(line);
	line = br.readLine();
	text.append("\n");
}
/* 5. Close BufferedReader stream */
br.close();
```

Να σημειωθεί ότι στο πρώτο βήμα που εκτελείται, γίνεται κλήση της συνάρτησης getFilepath() η οποία περιγράφεται στην κλάση RC4alg.

2.2.3 Υπομενού Save

Επιλέγοντας Save, εμφανίζεται παράθυρο διαλόγου μέσω του οποίου ο χρήστης είναι σε θέση να αποθηκεύσει το κείμενό του στο δίσκο του υπολογιστή ή σε κάποιο εξωτερικό μέσο αποθήκευσης. Ο πηγαίος κώδικας αυτού του υπομενού είναι:

```
/* Text file creation */
/* 1. "Save as" dialog box */
Frame frame = null;
FileDialog fDialog = new FileDialog(frame, "Save", FileDialog.SAVE);
fDialog.setVisible(true);
/* 2. Get the chosen filename and directory */
String filename = fDialog.getFile();
String directory = fDialog.getDirectory();
File file = new File(directory + filename + ".txt");
/* 3. Create the the file and write the content of the textarea in it */
FileWriter writer = null;
writer = new FileWriter(file);
String textarea = text.getText(); // save textarea content in plaintext
writer.write(textarea);
// 4. Close the file
writer.close(); // required !!!
```

2.2.4 Υπομενού Exit

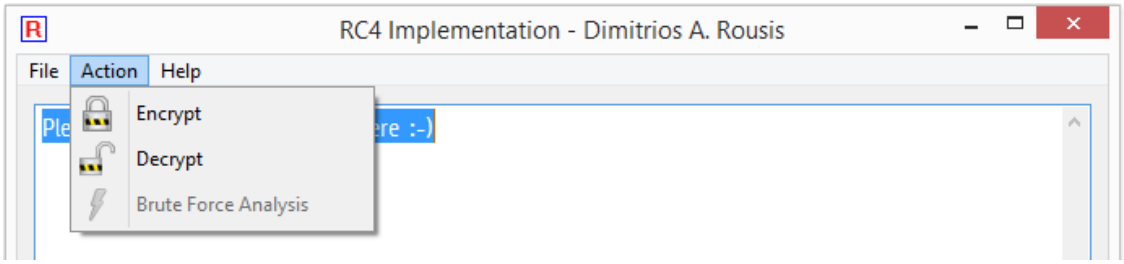
Τερματίζει την εφαρμογή. Ο πηγαίος κώδικας είναι:

```
System.exit(0);
```

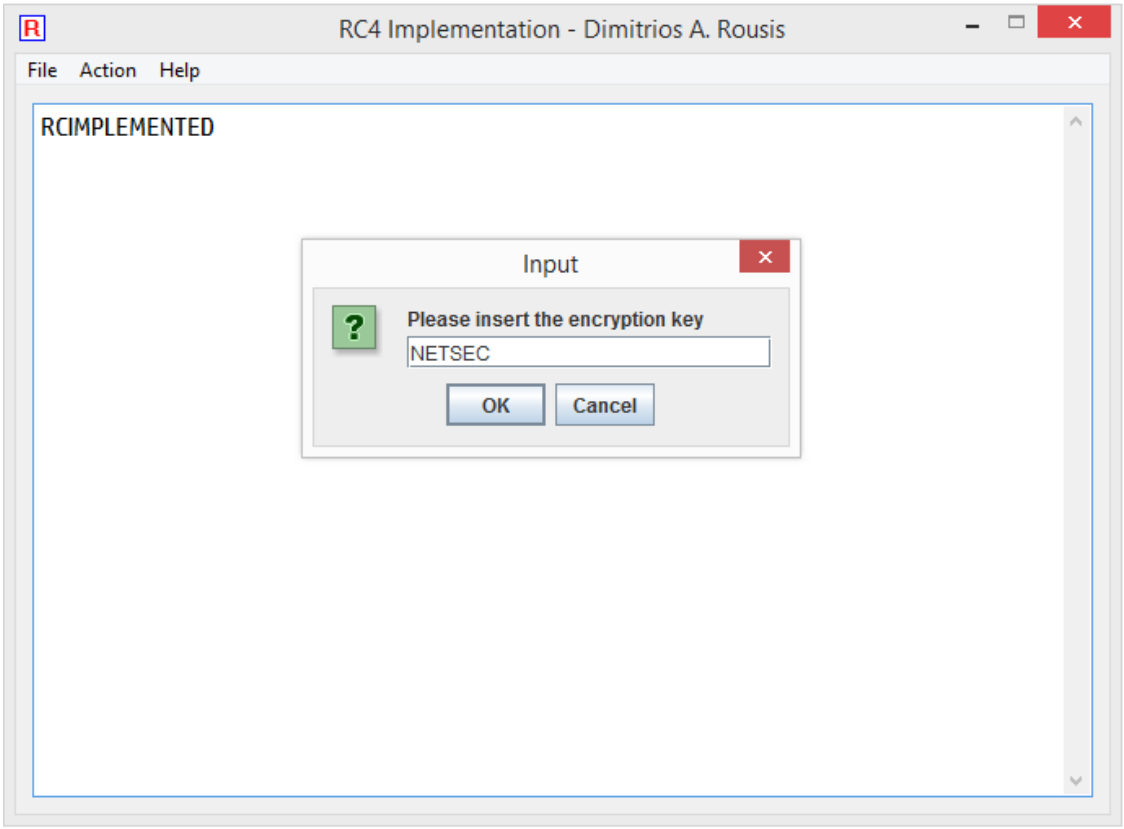
Το υπομενού Exit είναι και το τελευταίο του μενού File.

2.2.5 **Υπομενού** Encrypt

Το μενού Action όπως ήδη αναφέρθηκε είναι το σημαντικότερο μιας και σε αυτό συνοψίζεται ο λόγος ύπαρξης ολόκληρης της εφαρμογής.



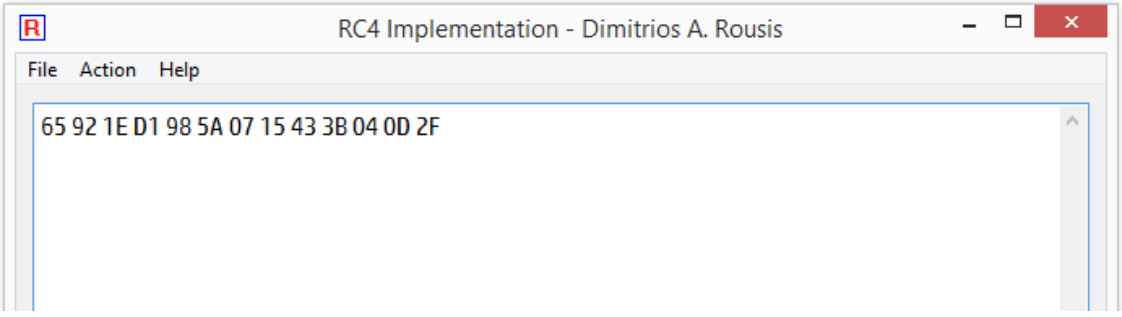
Το πρώτο υπομενού είναι το Encrypt. Επιλέγοντάς το, αρχικά εμφανίζεται ένα παράθυρο διαλόγου στο οποίο ο χρήστης έχει τη δυνατότητα να εισάγει το κλειδί κρυπτογράφησης:



Στη συνέχεια το κλειδί αυτό χρησιμοποιείται για την παραγωγή του κρυπτογραφημένου κειμένου το οποίο και παρουσιάζεται σε δεκαεξαδική πλέον μορφη, στην θέση του αρχικού. Ο πηγαίος κώδικας του υπομενού Encrypt είναι:

```
/* 1. Get the user's plaintext from the textarea */
String textarea = text.getText();
/* 2. Convert user's plaintext from the textarea to int array */
int plaintext[] = new int[textarea.length()];
for(int i=0; i<textarea.length(); i++)
{
	plaintext[i] = (int) textarea.charAt(i);
}
/* 3. Ask the user for the encryption key */
JOptionPane frame = new JOptionPane();
String userkey = JOptionPane.showInputDialog(frame, "Please insert the
                                                  encryption key");
/* 4. Convert the encryption key from String to int array */
int Key[]= new int[userkey.length()];
for(int i=0; i<userkey.length(); i++)
{
	Key[i] = (int) userkey.charAt(i);
}
/* 5. Pass the key to RC4alg class constructor */
RC4alg rc4 = new RC4alg(Key);
/* 6. Encrypt the plaintext */
int[] cipher = rc4.encrypt(plaintext);
/* 7. Convert the chiphertext from int array to HEX */
String ciphertext="";
for(int i=0;i<cipher.length;i++)
{
	ciphertext = ciphertext + Integer.toHexString(cipher[i])+" ";
}
/* 8. Set the textarea with the encrypted text */
text.setText(ciphertext);
```

Ενδεικτικά, η έξοδος της κρυπτογράφησης του RC4 για μυστικό κλειδί NETSEC και plaintext RCIMPLEMENTED είναι 65 92 1E D1 98 5A 07 15 43 3B 04 0D 2F σε δεκαεξαδική μορφή:



2.2.6 **Υπομενού** Decrypt

Το υπομενού Decrypt εκτελεί την διαδικασία της αποκρυπτογράφησης. Επιλέγοντάς το, εμφανίζεται ένα παράθυρο διαλόγου στο οποίο ο χρήστης πρέπει να εισάγει το κλειδί αποκρυπτογράφησης. Δηλαδή το ίδιο κλειδί που χρησιμοποίησε και για την κρυπτογράφηση του ίδιου κειμένου. Στη συνέχεια το κλειδί αυτό χρησιμοποιείται για την παραγωγή του αρχικού, μη-κρυπτογραφημένου κειμένου το οποίο παρουσιάζεται και πάλι στο πλαίσιο κειμένου της εφαρμογής. Ο πηγαίος κώδικας του υπομενού Decrypt είναι:

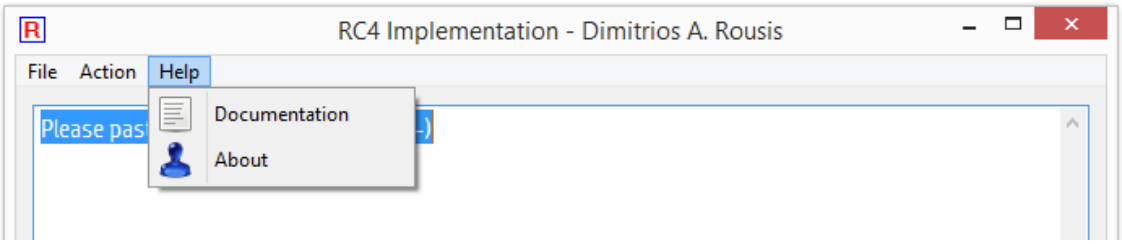
```
/* 1. Get the user's ciphertext from the textarea */
String textarea = text.getText();
String splitedciphertext[] = textarea.split(" ");
/* 2. Convert user's ciphertext from String to int array */
int ciphertext[] = new int[splitedciphertext.length];
for(int i=0; i<ciphertext.length; i++)
{
	ciphertext[i] = Integer.parseInt(splitedciphertext[i], 16) & 0xFF;
}
/* 3. Ask the user for the decryption key */
JOptionPane frame = new JOptionPane();
String userkey = JOptionPane.showInputDialog(frame, "Please insert the
                                                    decryption key");
/* 4. Convert the decryption key from String to int array */
int Key[]= new int[userkey.length()];
for(int i=0; i<userkey.length(); i++)
{
	Key[i] = (int) userkey.charAt(i);
}
/* 5. Pass the key to RC4alg class constructor */
RC4alg rc4 = new RC4alg(Key);
/* 6. Decrypt the ciphertext */
int intplaintext[] = rc4.decrypt(ciphertext);
/* 7. Convert integers to ASCII */
String plaintext="";
for(int i=0;i<intplaintext.length;i++)
{
	plaintext = plaintext + Character.toString((char) intplaintext[i]);
}
/* 8 Set the textarea with the plaintext */
text.setText(plaintext);
```

2.2.7 **Υπομενού** Brute Force Analysis

(δεν υποστηρίζεται σε αυτή την έκδοση)

2.2.8 **Υπομενού** Documentation

Το υπομενού Documentation είναι το πρώτο από τα δύο που περιλαμβάνονται στο μενού Help:



Ο ρόλος του είναι να παρέχει την απαραίτητη τεκμηρίωση της εφαρμογής που στην προκειμένη περίπτωση είναι ένα πλήρες αντίγραφο του παρόντος εγγράφου σε μορφή PDF. Ο πηγαίος κώδικας του υπομενού Documentation είναι:

```
File myFile = new File("1.pdf");
Desktop.getDesktop().open(myFile);
```

2.2.9 **Υπομενού** About

Το υπομενού About είναι βοηθητικό και όπως είναι αναμενόμενο, σκοπός του είναι να παρέχει πληροφορίες που αφορούν την εφαρμογή. Επιλέγοντάς το, ανοίγει ο προεπιλεγμένος browser του τρέχοντος λειτουργικού συστήματος και στη συνέχεια εμφανίζεται η ιστοσελίδα υποστήριξης της εφαρμογής (www.mythesis.org/rc4-encryption-algorithm-in-java).

Στον ιστότοπο www.MyThesis.gr θα είναι διαθέσιμος ολόκληρος ο πηγαίος κώδικας της εφαρμογής, Η τεκμηρίωσή της, καθώς και το εκτελέσιμό της. Με αυτό τον τρόπο θα μπορεί να λειτουργεί σαν ένα παράδειγμα υλοποίησης του RC4 σε γραφικό περιβάλλον για φοιτητές οι οποίοι ενδιαφέρονται. Επίσης, πιθανές βελτιώσεις και νέες εκδόσεις της εφαρμογής, θα είναι άμεσα διαθέσιμες για λήψη. Τέλος θα δίνεται η δυνατότητα Feedback και επίλυσης προβλημάτων.

Ο πηγαίος κώδικας του υπομενού About είναι:

```
String URL = "www.mythesis.org/rc4-encryption-algorithm-in-java";
Desktop.getDesktop().browse(URI.create(URL));
```

3. Οδηγίες ασφαλούς χρήσης

Δυστυχώς ο διαθέσιμος χρόνος για την ανάπτυξη μιας εφαρμογής στα πλαίσια ενός ακαδημαϊκού εξαμήνου είναι πολύ λίγος για την υλοποίηση μιας πλήρους εφαρμογής με γραφικό περιβάλλον και τεκμηρίωση.

Η εφαρμογή έχει πολλά bugs τα οποία όμως είναι γνωστά και πρόκειται να διορθωθούν σε επόμενη έκδοση. Παρ' όλα αυτά, η ζητούμενη λειτουργικότητα είναι πλήρως διαθέσιμη. Ο χρήστης μπορεί να επικολλήσει το κείμενό του στο αντίστοιχο πεδίο και στη συνέχει να εκτελέσει τις κύριες λειτουργίες που είναι η κρυπτογράφηση και η αποκρυπτογράφηση.

Όσον αφορά όμως την πλήρη χρήση του γραφικού περιβάλλοντος υπάρχουν κάποιοι περιορισμοί που θα πρέπει να ακολουθούνται προκειμένου η εφαρμογή να μην τερματίζεται απροσδόκητα:

1. Οι χαρακτήρες του plaintext θα πρέπει να είναι λατινικοί
2. Όταν επιλέγεται το υπομενού New από το μενού File, δεν υπάρχει προειδοποίηση για την αποθήκευση του ήδη πληκτρολογημένου κειμένου (πχ: θα θέλατε να σώσετε το κείμενο σας πριν δημιουργήσετε νέο ?). Όλα τα περιεχόμενα του πεδίου κειμένου σβήνονται άμεσα και ανοίγει ένα νέο.
3. Τα αρχεία που μπορούν να κρυπτογραφηθούν είναι μόνο αρχεία κειμένου. Παρ' όλα αυτά δεν υπάρχει κάποια προειδοποίηση κατά το άνοιγμα ή την αποθήκευση μιας μορφής αρχείου που δεν υποστηρίζεται.

4. Μελλοντικές βελτιώσεις και αμυντικός προγραμματισμός

Στην προηγούμενη παράγραφο αναφέρθηκαν οι διαδικασίες χρήσης της εφαρμογής προκειμένου αυτή να μην τερματίζεται απροσδόκητα. Οι περισσότεροι περιορισμοί προκύπτουν από την έλλειψη ενσωμάτωσης του λεγόμενου αμυντικού προγραμματισμού στον κώδικα. Ουσιαστικά πρόκειται για μεθόδους χειρισμού εξαιρέσεων και βρόχους ελέγχου που φροντίζουν από τη μία να μην εκτελούνται λανθασμένες ενέργειες και από την άλλη, εάν κάποια από αυτές συμβεί, προσδίδουν ανοχή στα σφάλματα που παράγονται.

Επίσης στην ίδια παράγραφο αναφέρεται ο περιορισμένος διαθέσιμος χρόνος για την ανάπτυξη της εφαρμογής. Αυτό δεν έχει μόνο επιπτώσεις στην αδυναμία εξάλειψης σφαλμάτων του

κώδικα αλλά και στην απουσία κάποιων χαρακτηριστικών που θα έκαναν την εφαρμογή πιο εύχρηστη και αποτελεσματική. Τα χαρακτηριστικά αυτά θα μπορούσαν να αποτελέσουν μελλοντικές βελτιώσεις και τα κυριότερα συνοψίζονται πιο κάτω:

- ✓ PRNG (Pseudorandom number generator) [βλ. θεωρητικό μέρος]
- ✓ BFA (Bute force analysis)
- ✓ Υποστήριξη για διαφορετικές κωδικοποιήσεις όπως UTF-8 κλπ

5. Προσωπικές απόψεις και συμπεράσματα

Αναμφισβήτητα η επιστήμη της κρυπτογραφίας παράγει σημαντικό και δύσκολο έργο για τις τρέχουσες μεθόδους επικοινωνίας. Πολύπλοκοι μαθηματικοί υπολογισμοί δημιουργούν έξυπνους αλγόριθμους οι οποίοι θωρακίζουν ευαίσθητα δεδομένα από ανεπιθύμητη πρόσβαση.

Παρ' όλα αυτά ένας αλγόριθμος κρυπτογράφησης ο οποίος εκτελείται σε ηλεκτρονικό υπολογιστικό σύστημα, σχεδόν πάντα, υλοποιείται σε κάποια γλώσσα προγραμματισμού. Οι ιδιαιτερότητες της γλώσσας αυτής είναι μία πολύ σημαντική πρόκληση για την ορθή εφαρμογή του ψευδοκώδικα που "εύκολα" αναγράφεται στο έγγραφο της θεωρητικής περιγραφής του προς υλοποίηση κρυπταλγόριθμου.

Πιο συγκεκριμένα, κατά την υλοποίηση του RC4 στη Java, έρχεται κανείς αντιμέτωπος με ποικίλες δυσκολίες που μπορεί να προκύπτουν από το πιο απλό, όπως η σωστή προσπέλαση ενός αρχείου στο δίσκο, έως και πιο πολύπλοκα θέματα, όπως η κωδικοποίηση των χαρακτήρων που εισάγει ο χρήστης. Δεν ήταν λίγες οι φορές κατά την ανάπτυξη της παρούσας εφαρμογής που ένα συγκεκριμένο πρόβλημα έμενε άλυτο για μέρες.

Η προγραμματιστική εμπειρία για την εφαρμογή ενός αλγόριθμου κρυπτογράφησης είναι ιδιαίτερα σημαντική και επηρεάζει την ορθότητα του όλου εγχειρήματος. Ένα παράδειγμα αυτού αποτελούν σε πολλές περιπτώσεις και τα λεγόμενα "κενά ασφαλείας" τα οποία εκμεταλλεύονται με κακόβουλο τρόπο ορισμένοι χρήστες για να αποκτήσουν πρόσβαση σε απόρρητο περιεχόμενο.

Συμπερασματικά, η μελέτη της λογικής και μαθηματικής θεωρίας ενός κρυπταλγόριθμου μπορεί να είναι σημαντική, όμως εξίσου σημαντική είναι και η προγραμματιστική προσέγγιση.

Αυτό θα έπρεπε να λαμβάνεται πολύ σοβαρά υπόψη κατά τη διδασκαλία της κρυπτογραφίας στα πανεπιστημιακά και τεχνολογικά εκπαιδευτικά ιδρύματα. Η καλή κατανόηση και αποστήθιση τύπων και ορισμών, από μόνα τους, δεν οδηγούν απαραιτήτως και σε ασφαλείς εφαρμογές.

ΠΗΓΕΣ & ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Σημειώσεις του μαθήματος "Προχωρημένα Θέματα Ασφάλειας", ΠΜΣ Πληροφορικής ΑΠΘ.
2. https://github.com/oxee/RC4/blob/master/RC4.java
3. http://en.wikipedia.org/wiki/RC4
4. http://en.wikipedia.org/wiki/Stream_cipher
5. http://en.wikipedia.org/wiki/RSA_Security

ΠΑΡΑΡΤΗΜΑΤΑ

Α. Ο πηγαίος κώδικας του αρχείου RC4alg.java

```
package rc4;

import java.awt.FileDialog;
import java.awt.Frame;

public class RC4alg
{
    private int S[] = new int[256];
    private int T[] = new int[256];

    /* Main Constructor - Init function */
    public RC4alg(int key[])
    {
        int j, swap;

        /* Initialization */
        for(int i=0; i<256; i++)
        {
           S[i]= i;
           T[i]= key[i % key.length];
        }

        /* Initial permutation of S */
        j = 0;
        for(int i=0; i<256; i++)
        {
           j=(j + S[i] + T[i]) % 256;
           /* classic swap */
           swap = S[i];
           S[i] = S[j];
           S[j] = swap;
        }
    }

    /* RC4 encrypt function */
    public int[] encrypt(int plaintext[])
    {
       int ciphertext[];
       ciphertext = new int[plaintext.length];
       int i, j, swap, t;

       i = j = 0 ;
       for(int count=0; count<plaintext.length; count++)
       {
          i = (i + 1) % 256;
          j = (j + S[i]) % 256;
```

```
            /* classic swap */
            swap = S[i];
            S[i] = S[j];
            S[j] = swap;

            t = (S[i] + S[j]) % 256;

            ciphertext[count] = plaintext[count] ^ S[t];
         }
         return ciphertext;
      }

      /* Decrypt function */
      public int[] decrypt(int cipher[])
      {
         return encrypt(cipher);
      }

      /* File pathname discovery function */
      public static String getFilepath()
      {
         /* Load file dialog box */
         Frame frame = null;
               FileDialog fDialog = new FileDialog(frame, "Open",
FileDialog.LOAD);
          fDialog.setVisible(true);
          /* Calculate filename and directory of the file */
          String filename = fDialog.getFile();
          String directory = fDialog.getDirectory();
          /* Concatenate filename and directory into pathname */
          String path = directory + filename;
          /* Return the string of the pathname for further use */
         return path;
      }
}
```

Β. Ο πηγαίος κώδικας του αρχείου RC4.java

```
package rc4;

import java.awt.Desktop;
import java.awt.FileDialog;
import java.awt.Frame;
import java.awt.HeadlessException;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.net.URI;

import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.SWT;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;
import org.eclipse.swt.widgets.Menu;
import org.eclipse.swt.widgets.MenuItem;
import org.eclipse.wb.swt.SWTResourceManager;
import org.eclipse.swt.widgets.Text;

import java.io.IOException;

import javax.swing.JOptionPane;

//import com.sun.org.apache.xalan.internal.xsltc.compiler.Pattern;
//import com.sun.org.apache.xerces.internal.impl.xs.identity.Selector.Matcher;

public class RC4 {

	protected Shell shlRcImplementation;

	private Text text;

	/**
	 * @wbp.nonvisual location=182,209
	 */

	// private final FormToolkit formToolkit = new FormToolkit(Display.getDefault());

	/**
	 * Launch the application.
	 * @param args
	 * @throws IOException
	 */
	public static void main(String[] args) throws IOException {
		try {
			RC4 window = new RC4();
			window.open();
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
```

```
/**
 * Open the window.
 */
public void open() {
	Display display = Display.getDefault();
	createContents();
	shlRcImplementation.open();
	shlRcImplementation.layout();
	while (!shlRcImplementation.isDisposed()) {
		if (!display.readAndDispatch()) {
			display.sleep();
		}
	}
}


/**
 * Create contents of the window.
 */
protected void createContents()
{
	shlRcImplementation = new Shell();
	shlRcImplementation.setModified(true);
	shlRcImplementation.setFullScreen(true);
	shlRcImplementation.setImage(SWTResourceManager.getImage(RC4.class,
                                                "/images/rousis.png"));
	shlRcImplementation.setSize(687, 505);
	shlRcImplementation.setText("RC4 Implementation - Dimitrios A. Rousis");
	shlRcImplementation.setMinimumSize(687, 505);
	shlRcImplementation.setLayout(null);

	/* Create textarea when the app starts */
	text = new Text(shlRcImplementation, SWT.BORDER | SWT.WRAP | SWT.V_SCROLL
                                                        | SWT.MULTI);
	text.setText("Please paste your precious text here  :-)");
	text.setFont(SWTResourceManager.getFont("HP Simplified", 11, SWT.NORMAL));
	text.setBounds(10, 10, 651, 426);
	text.selectAll();

	/* Menu Bar - Contains all menus */
	Menu menu = new Menu(shlRcImplementation, SWT.BAR);
	shlRcImplementation.setMenuBar(menu);

	/* Start of Cascade Menu "File" - Contains "New", "Open", "Save" and
                                                        "Exit" MenuItems */
	MenuItem cascadeMenuFile = new MenuItem(menu, SWT.CASCADE);
	cascadeMenuFile.setText("File");
	Menu menuFile = new Menu(cascadeMenuFile);
	cascadeMenuFile.setMenu(menuFile);

	/* MenuItem: New */
	MenuItem New = new MenuItem(menuFile, SWT.NONE);
	New.setImage(SWTResourceManager.getImage(RC4.class, "/images/new.png"));
	New.setText("New");
	New.addSelectionListener(new SelectionAdapter()
	{
		public void widgetSelected(SelectionEvent e)
		{
			try
			{
				text.setText("Please paste your precious text here
                                                                :-)");
				text.selectAll();
			}
			catch (Exception e1)
			{
				e1.printStackTrace();
```

```
                }
        }
}
);

/* MenuItem: Open */
MenuItem Open = new MenuItem(menuFile, SWT.NONE);
Open.setImage(SWTResourceManager.getImage(RC4.class, "/images/open.png"));
Open.setText("Open");
Open.addSelectionListener(new SelectionAdapter()
{
        public void widgetSelected(SelectionEvent e)
        {
                try
                {
                /* 1. Find out the pathname of */
                 String filepath = RC4alg.getFilepath();
                /* 2. Clear the contents of the textaarea */
                 text.setText("");
                 /* 3. Read the contents of the text file opened */
                BufferedReader br = new BufferedReader(new
                                        FileReader(filepath));
                String text1 = "";
                String line = br.readLine();
                        /* 4. Print the contents of the text file to the
                                                                text area */
                int len = line.length();
                for(int i=0; i<len; i++)
                {
                        text.append(line);
                        line = br.readLine();
                        text.append("\n");
                }
                /* 5. Close BufferedReader stream */
                br.close();
                }
                catch (Exception e1)
                {
                        e1.printStackTrace();
                }
        }
}
);


/* MenuItem: Save */
MenuItem Save = new MenuItem(menuFile, SWT.NONE);
Save.setImage(SWTResourceManager.getImage(RC4.class, "/images/save.png"));
Save.setText("Save");
Save.addSelectionListener(new SelectionAdapter()
{
        public void widgetSelected(SelectionEvent e)
        {
                try
                {
                /* Text file creation */
                /* 1. "Save as" dialog box */
                Frame frame = null;
                FileDialog fDialog = new FileDialog(frame, "Save",
                                                        FileDialog.SAVE);
                 fDialog.setVisible(true);
                 /* 2. Get the chosen filename and directory */
                 String filename = fDialog.getFile();
                 String directory = fDialog.getDirectory();
                 File file = new File(directory + filename + ".txt");
```

```
                /* 3. Create the the file and write the content of the
                                                        textarea in it */
                FileWriter writer = null;
                writer = new FileWriter(file);
                String textarea = text.getText();
                /* save textarea content in plaintext */
                writer.write(textarea);
                /* 4. Close the file */
                writer.close(); // required !!!
                }
                catch (Exception e1)
                {
                        e1.printStackTrace();
                }
        }
}
);

/* Separator */
new MenuItem(menuFile, SWT.SEPARATOR);

/* MenuItem: Exit */
MenuItem Exit = new MenuItem(menuFile, SWT.NONE);
Exit.setImage(SWTResourceManager.getImage(RC4.class,
                                "/images/close.png"));
Exit.setText("Exit");
Exit.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent e) {
                System.exit(0);
        }
});
/* End of Cascade Menu "File" */


/* Start of Cascade Menu "Action" - Contains "Encrypt", "Decrypt" and
                                        "Brute Force Analysis" MenuItems */
MenuItem cascadeMenuAction = new MenuItem(menu, SWT.CASCADE);
cascadeMenuAction.setText("Action");
Menu menuAction = new Menu(cascadeMenuAction);
cascadeMenuAction.setMenu(menuAction);

/* MenuItem: Encrypt */
MenuItem Encrypt = new MenuItem(menuAction, SWT.NONE);
Encrypt.setImage(SWTResourceManager.getImage(RC4.class,
                                        "/images/encrypt.png"));
Encrypt.setText("Encrypt");
Encrypt.addSelectionListener(new SelectionAdapter() {
        public void widgetSelected(SelectionEvent e) {
                /* 1. Get the user's plaintext from the textarea */
                String textarea = text.getText();
                /* 2. Convert user's plaintext from the textarea to int
                                                                array */
                int plaintext[] = new int[textarea.length()];
                for(int i=0; i<textarea.length(); i++)
                {
                        plaintext[i] = (int) textarea.charAt(i);
                }
                /* 3. Ask the user for the encryption key */
                JOptionPane frame = new JOptionPane();
                String userkey = JOptionPane.showInputDialog(frame, "Please
                                                insert the encryption key");
```

```
		/* 4. Convert the encryption key from String to int array */
			int Key[]= new int[userkey.length()];
			for(int i=0; i<userkey.length(); i++)
			{
				Key[i] = (int) userkey.charAt(i);
			}
			/* 5. Pass the key to RC4alg class constructor */
			RC4alg rc4 = new RC4alg(Key);
			/* 6. Encrypt the plaintext */
			int[] cipher = rc4.encrypt(plaintext);
			/* 7. Convert the chiphertext from int array to HEX */
			String ciphertext="";
			for(int i=0;i<cipher.length;i++)
			{
				ciphertext = ciphertext +
				Integer.toHexString(cipher[i])+" ";
			}
			/* 8. Set the textarea with the encrypted text */
			text.setText(ciphertext);
		}
});

/* MenuItem: Decrypt */
MenuItem Decrypt = new MenuItem(menuAction, SWT.NONE);
Decrypt.setImage(SWTResourceManager.getImage(RC4.class,
                                    "/images/decrypt.png"));
Decrypt.setText("Decrypt");
Decrypt.addSelectionListener(new SelectionAdapter() {
	public void widgetSelected(SelectionEvent e) {
			try
			{
			/* 1. Get the user's ciphertext from the textarea */
			String textarea = text.getText();
			String splitedciphertext[] = textarea.split(" ");
			/* 2. Convert user's ciphertext from String to int
                                                    array */
			int ciphertext[] = new
			int[splitedciphertext.length];
			for(int i=0; i<ciphertext.length; i++)
			{
				ciphertext[i] =
				Integer.parseInt(splitedciphertext[i], 16) & 0xFF;
			}
			/* 3. Ask the user for the decryption key */
			JOptionPane frame = new JOptionPane();
			String userkey = JOptionPane.showInputDialog(frame,
                        "Please insert the decryption key");
			/* 4. Convert the decryption key from String to int
                                                    array */
			int Key[]= new int[userkey.length()];
			for(int i=0; i<userkey.length(); i++)
			{
				Key[i] = (int) userkey.charAt(i);
			}
			/* 5. Pass the key to RC4alg class constructor */
			RC4alg rc4 = new RC4alg(Key);
			/* 6. Decrypt the ciphertext */
			int intplaintext[] = rc4.decrypt(ciphertext);
			/* 7. Convert integers to ASCII */
			String plaintext="";
			for(int i=0;i<intplaintext.length;i++)
			{
				plaintext = plaintext +
				Character.toString((char) intplaintext[i]);
			}
```

```
                /* 8 Set the textarea with the plaintext */
                text.setText(plaintext);
                } catch (NumberFormatException e1) {
                        // TODO Auto-generated catch block
                        e1.printStackTrace();
                } catch (HeadlessException e1) {
                        // TODO Auto-generated catch block
                        e1.printStackTrace();
                }
        }
});

/* MenuItem: Brute Force Analysis */
MenuItem BruteForceAnalysis = new MenuItem(menuAction, SWT.NONE);
BruteForceAnalysis.setImage(SWTResourceManager.getImage(RC4.class,
                                                "/images/brute.png"));
BruteForceAnalysis.setText("Brute Force Analysis");
BruteForceAnalysis.setEnabled(false); // delete this when BFA is
                                                        implemented
/* End of Cascade Menu "Action" */

/* Start of Cascade Menu "Help" - Contains "Documentation", and "About"
                                                                MenuItems */
MenuItem cascadeMenuHelp = new MenuItem(menu, SWT.CASCADE);
cascadeMenuHelp.setText("Help");
Menu menuHelp = new Menu(cascadeMenuHelp);
cascadeMenuHelp.setMenu(menuHelp);

/* MenuItem: Documentation */
MenuItem Documentation = new MenuItem(menuHelp, SWT.NONE);
Documentation.setImage(SWTResourceManager.getImage(RC4.class,
                                "/images/documentation.png"));
Documentation.setText("Documentation");
Documentation.addSelectionListener(new SelectionAdapter()
{
        public void widgetSelected(SelectionEvent e)
        {
                try
                {
                        File myFile = new File("1.pdf");
                 Desktop.getDesktop().open(myFile);
                }
                catch (Exception e1)
                {
                        e1.printStackTrace();
                }
        }
}
);
```

```
                /* MenuItem: About */
                MenuItem About = new MenuItem(menuHelp, SWT.NONE);
                About.setImage(SWTResourceManager.getImage(RC4.class,
                                                "/images/about.png"));
                About.setText("About");
                About.addSelectionListener(new SelectionAdapter()
                {
                        public void widgetSelected(SelectionEvent e)
                        {
                                try
                                {
                                        String URL = "www.mythesis.org/rc4-encryption-
                                                                        algorithm-in-java";
                                        Desktop.getDesktop().browse(URI.create(URL));
                                }
                                catch (Exception e1)
                                {
                                        e1.printStackTrace();
                                }
                        }
                }


                );
                /* End of Cascade Menu "Help" */

                /* End of Menu Bar - Contains all menus */
        }
}
```