Cahier de programmation R.S.A.

Delbot Francois Rovedakis Stephane

5 mai 2005

Table des matières

Chapitre 1

Liste complète des fonctions de l'application

1.1 About

1.1.1 Prototype de la fonction

LRESULT CALLBACK About (HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

1.1.2 Paramètres

en entrée

- HWND hDlg : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.1.3 Description

Cette fonction gère la boucle des messages associés à la boîte de dialogue About. La boîte de dialogue About contient des informations sur la version de l'application, ses auteurs...

1.2 affiche_barre_progression

1.2.1 Prototype de la fonction

void affiche_barre_progression (HWND barre, BOOL bVisible)

1.2.2 Paramètres

en entrée

- HWND barre : L'identifiant de la barre de progression.
- BOOL bVisible : Booleen indiquant si la barre de progression doit être masquée (False) ou affichée (True).

en sortie

- void : Rien.

1.2.3 Description

Cette fonction permet d'afficher ou de masquer une barre de progression. Cela permet de ne l'afficher que lorsqu'une opération est en cours d'exécution.

1.3 chiffrer

1.3.1 Prototype de la fonction

void chiffrer (HWND hWnd, long identifiant, char *source, char *destination, mpz_t n, mpz_t e)

1.3.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue appelant la fonction de chiffrement.
- long identifiant : Identifiant de la barre de progression.
- char *source : Le chemin du fichier que l'on souhaite chiffrer.
- char *destination : Le chemin du fichier résultant du chiffrement du fichier source.
- mpz_t n : Entier n de la clé publique (n,e).
- mpz_t e : Entier e de la clé publique (n,e).

en sortie

- void : Rien.

1.3.3 Description

Cette fonction permet de chiffrer par RSA un fichier source et d'enregistrer le résultat dans un autre fichier en utilisant une clé publique (n,e).

1.4 chiffrerdlg

1.4.1 Prototype de la fonction

LRESULT CALLBACK chiffrerdlg (HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

1.4.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.4.3 Description

Cette fonction gère la boucle des messages associés à la boîte de dialogue chiffrerdlg. La boîte de dialogue chiffrerdlg permet de chiffer un document, charger un fichier de clé publique. Cette boîte de dialogue possède une barre de progression pour suivre l'état d'avancement du chiffrement.

1.5 clersa

1.5.1 Prototype de la fonction

LRESULT CALLBACK clersa (HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

1.5.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.5.3 Description

Cette fonction gère la boucle des messages associés à la boîte de dialogue clersa. La boîte de dialogue clersa permet la création et l'enregistrement de clés RSA.

1.6 compter_nombre_dechiffrer

1.6.1 Prototype de la fonction

long compter_nombre_dechiffrer (char *source)

1.6.2 Paramètres

en entrée

- char *source : Chemin d'un fichier chiffré par RSA.

en sortie

- long : Le nombre de blocs (d'entiers) du fichier.

1.6.3 Description

Cette fonction permet de connaître le nombre de blocs chiffrés par RSA. Cela permet de connaitre le nombre d'étapes nécessaires pour l'opération de déchiffrement, et ainsi de mettre la barre de progression du déchiffrement à jour.

1.7 dechiffrer

1.7.1 Prototype de la fonction

void dechiffrer (HWND hWnd, long identifiant, char *source, char *destination, mpz_t n, mpz_t d)

1.7.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue appelant la fonction de déchiffrement.
- long identifiant : Identifiant de la barre de progression.
- char *source : Le chemin du fichier que l'on souhaite déchiffrer.
- char *destination : Le chemin du fichier résultant du déchiffrement du fichier source.
- mpz_t n : Entier n de la clé privée (n,d).
- mpz_t e : Entier e de la clé privée (n,d).

en sortie

- void : Rien.

1.7.3 Description

Cette fonction permet de déchiffrer un fichier précedemment crypté par RSA et d'enregistrer le résultat dans un autre fichier en utilisant une clé privée (n,d).

1.8 dechiffrerdlg

1.8.1 Prototype de la fonction

LRESULT CALLBACK dechiffrerdlg (HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

1.8.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.8.3 Description

Cette fonction gère la boucle des messages associés à la boîte de dialogue dechiffrerdlg. La boîte de dialogue dechiffrerdlg permet de déchiffer un document, charger un fichier de clé privée. Cette boîte de dialogue possède une barre de progression pour suivre l'état d'avancement du déchiffrement.

1.9 ecrire_mpz

1.9.1 Prototype de la fonction

void ecrire_mpz (FILE *fp, mpz_t b, long nbcl)

1.9.2 Paramètres

en entrée

- FILE *fp : Pointeur vers le fichier de destination.
- mpz_t b : Un entier b que l'on souhaite écrire dans le fichier de destination.
- long nbcl : Taille de l'entier b avant chiffrement par RSA.

en sortie

- void : Rien.

1.9.3 Description

Cette fonction permet d'écrire un entier dans un fichier. La structure d'enregistrement choisie est la suivante :
- nombre de digits de l'entier b avant chiffrement.
- nombre de digits de l'entier b après chiffrement.
- chaine de caractères representant l'entier b après chiffrement.

1.10 finalisation_barre_progression

1.10.1 Prototype de la fonction

void finalisation_barre_progression (HWND hWnd, long identifiant)

1.10.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la fenêtre parente de la barre de progression.
- long identifiant : Identifiant de la barre de progression.

en sortie

- void : Rien.

1.10.3 Description

Cette fonction permet de masquer et de réinitialiser une barre de progression à la fin d'une opération.

1.11 generer_d

1.11.1 Prototype de la fonction

void generer_d (mpz_t d, mpz_t e, mpz_t psi_n)

1.11.2 Paramètres

en entrée

- mpz_t d : Pointeur vers un entier d.
- mpz_t e : Pointeur vers un entier e.
- mpz_t psi_n : Pointeur vers un entier psi_n.

en sortie

- void : Rien.

1.11.3 Description

Cette fonction calcule la valeur de l'entier d en fonction de e et de psi_n, grâce à la formule suivante : $d = e^{-1} (\text{mod } \varphi(\text{n}))$.

1.12 generer_e

1.12.1 Prototype de la fonction

void generer_e (mpz_t e, mpz_t psi_n)

1.12.2 Paramètres

en entrée

- mpz_t e : Pointeur vers un entier e.
- mpz_t psi_n : Pointeur vers un entier psi_n.

en sortie

- void : Rien.

1.12.3 Description

Cette fonction génère un entier et l'enregistre dans e tel que PGCD(e,φ(n))=1 avec $1 < e < \varphi(n)$.

1.13 generer_une_cle_RSA_aleatoire

1.13.1 Prototype de la fonction

void generer_une_cle_RSA_aleatoire (mpz_t p, mpz_t q, mpz_t e, mpz_t d, mpz_t n, mpz_t psi_n)

1.13.2 Paramètres

en entrée

- mpz_t p : Pointeur vers un entier p.
- mpz_t q : Pointeur vers un entier q.
- mpz_t e : Pointeur vers un entier e.
- mpz_t d : Pointeur vers un entier d.
- mpz_t n : Pointeur vers un entier n.
- mpz_t psi_n : Pointeur vers un entier psi_n.

en sortie

- void : Rien.

1.13.3 Description

Cette fonction génère une clé RSA valide :

1. Elle génère deux nombres premiers qu'elle stocke dans p et q.
2. Elle calcule pq et stocke le résultat dans n.
3. Elle calcule (p-1)(q-1) et stocke le résultat dans psi_n.
4. Elle génère un entier et l'enregistre dans e tel que PGCD(e,φ(n))=1 avec $1 < e < \varphi(n)$.
5. Elle calcule un entier en fonction de e et de psi_n et le stocke dans d, grâce à la formule suivante : $d = e^{-1} (\text{mod } \varphi(\text{n}))$.

1.14 incremente_barre_progression

1.14.1 Prototype de la fonction

void incremente_barre_progression (HWND hWnd, long identifiant)

1.14.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la fenêtre parente de la barre de progression.
- long identifiant : Identifiant de la barre de progression.

en sortie

- void : Rien.

1.14.3 Description

Cette fonction fait avancer la barre de progression d'un cran.

1.15 initialisation_barre_progression

1.15.1 Prototype de la fonction

void initialisation_barre_progression (HWND hWnd, long max, long identifiant)

1.15.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la fenêtre parente de la barre de progression.
- long max : Le nombre d'étapes maximum lors de la progression de la barre.
- long identifiant : Identifiant de la barre de progression.

en sortie

- void : Rien.

1.15.3 Description

Cette fonction initialise une barre de progression en lui affectant une borne max et en mettant sa progression initiale à 0.

1.16 initrandom

1.16.1 Prototype de la fonction

void initrandom (void)

1.16.2 Paramètres

en entrée

- void : Rien

en sortie

- void : Rien.

1.16.3 Description

Cette fonction initialise la génération de grands nombres aléatoires.

1.17 lire_mpz

1.17.1 Prototype de la fonction

long lire_mpz (FILE *fp, mpz_t b)

1.17.2 Paramètres

en entrée

- FILE *fp : Pointeur de fichier à partir duquel on souhaite lire un entier
- mpz_t b : Pointeur vers l'entier dans lequel sera stocké l'entier lu.

en sortie

- long : Taille (en nombre de digits) de l'entier avant chiffrement par RSA.

1.17.3 Description

Cette fonction permet de lire un entier à partir d'un fichier source en respectant la structure :
– nombre de digits de l'entier b avant chiffrement.
– nombre de digits de l'entier b après chiffrement.
– chaine de caractères representant l'entier b après chiffrement.

1.18 trouver_un_nombre_premier

1.18.1 Prototype de la fonction

void trouver_un_nombre_premier (mpz_t a)

1.18.2 Paramètres

en entrée

- mpz_t a : Pointeur vers l'entier dans lequel sera stocké le résultat.

en sortie

- void : Rien.

1.18.3 Description

Cette fonction génère un nombre premier. Cette génération utilise le test de Miller-Rabin itéré 10 fois.

1.19 transformer_en_chaine

1.19.1 Prototype de la fonction

char * transformer_en_chaine (mpz_t b, long nbcl)

1.19.2 Paramètres

en entrée

- mpz_t b : Pointeur vers l'entier à transformer en chaîne de caractères.
- long nbcl : Taille finale de la chaîne de caractères.

en sortie

- char * : Pointeur vers la chaine de caractères résultat.

1.19.3 Description

Cette fonction transforme un entier en chaîne de caractères. Le paramètre nbcl nous indique la taille originale de l'entier(en nombre de digits). Si cette taille n'est pas atteinte, on rajoute des 0 en tête de la chaîne pour la completer. Ces 0 sont nécessaires, car chaque groupe de 3 caractères represente le code ASCII d'une lettre.

1.20 transformer_en_mpz

1.20.1 Prototype de la fonction

void transformer_en_mpz (mpz_t b, unsigned char *chaine, long taille)

1.20.2 Paramètres

en entrée

- mpz_t b : Pointeur vers l'entier dans lequel sera stocké le résultat.
- unsigned char *chaine : Chaîne de caractères que l'on souhaite convertir en entier.
- long taille : taille de la chaîne de caractères.

en sortie

- void : Rien.

1.20.3 Description

Cette fonction traduit une chaîne de caractères en un entier de la façon suivante : chaque caractère est convertit en son code ASCII concaténé à la suite. Par exemple la chaîne "abc" nous donne l'entier "097098099".

1.21 verification_validite_cle_RSA

1.21.1 Prototype de la fonction

int verification_validite_cle_RSA (mpz_t p, mpz_t q, mpz_t e, mpz_t d, mpz_t n, mpz_t psi_n)

1.21.2 Paramètres

en entrée

- mpz_t p : Pointeur vers un entier p.
- mpz_t q : Pointeur vers un entier q.
- mpz_t e : Pointeur vers un entier e.
- mpz_t d : Pointeur vers un entier d.
- mpz_t n : Pointeur vers un entier n.
- mpz_t psi_n : Pointeur vers un entier psi_n.

en sortie

- int : 0 si la clé est valide, 1 sinon.

1.21.3 Description

Cette fonction vérifie si une clé RSA est valide ou non.

1.22 WndProc

1.22.1 Prototype de la fonction

LRESULT CALLBACK WndProc (HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

1.22.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.22.3 Description

Cette fonction gère la boucle des messages associés à la Fenêtre principale. La fenêtre principale permet d'acceder aux différentes boites de dialogues de l'application (chiffrement d'un document, déchiffrement d'un document, signature et vérification de la provenance d'un document, génération de clés RSA...). La fenêtre principale affiche en permanence l'aide de l'application.

1.23 designerdlg

1.23.1 Prototype de la fonction

LRESULT CALLBACK designerdlg (HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

1.23.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.23.3 Description

Cette fonction gère la boucle des messages associés à la boîte de dialogue designerdlg. La boîte de dialogue designerdlg permet de vérifier qu'un document provient bien de la personne attendue. Cette boîte de dialogue possède une barre de progression pour suivre l'état d'avancement de cette vérification.

1.24 signerdlg

1.24.1 Prototype de la fonction

LRESULT CALLBACK signerdlg (HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

1.24.2 Paramètres

en entrée

- HWND hWnd : L'identifiant de la boite de dialogue.
- UINT message : Message évènementiel.
- WPARAM wParam : 1^{er} Paramètre du message.
- LPARAM lParam : 2^{eme} Paramètre du message.

en sortie

- LRESULT : Réponse spécifique au message reçu en entrée.

1.24.3 Description

Cette fonction gère la boucle des messages associés à la boîte de dialogue signerdlg. La boîte de dialogue signerdlg permet de signer un document, ainsi, tout le monde sera capable de lire ce document en étant sûr de sa provenance. Cette boîte de dialogue possède une barre de progression pour suivre l'état d'avancement de la signature du document.

1.25 mise_a_jour_TAILLEBLOC

1.25.1 Prototype de la fonction

void mise_a_jour_TAILLEBLOC (mpz_t n)

1.25.2 Paramètres

en entrée

- mpz_t n : Pointeur vers l'entier n d'une clé RSA.

en sortie

- void : Rien.

1.25.3 Description

Cette fonction calcule la taille maximum des blocs que nous pourrons chiffrer par RSA.

Chapitre 2

Les choix de programmation

2.1 Gestion des clés

Durant nos tests, nous nous sommes rendu compte qu'il était très génant de faire des copier/coller pour pouvoir réutiliser une clé RSA. Nous avons donc décider de créer un format de fichier permettant la sauvegarde des clés et leur chargement.

2.1.1 Génération de clés

Lorsque l'on génère une clé grâce à notre générateur de clé, il est possible de la sauvegarder. Cette sauvegarde créée deux fichiers :

- Un fichier .public, qui contiendra la clé publique, celle qui doit être publiée (n,e).
- Un fichier .private, qui contiendra la clé privée, qui doit rester secrete (n,d).

2.1.2 Les fichiers de clé privée

Les fichiers de clé privée sont organisés de la façon suivante :
"N :" chaine de caractères representant le module n en base 10.
"D :" chaine de caractères representant l'exposant d en base 10.

2.1.3 Les fichiers de clé publique

Les fichiers de clé publique sont organisés de la façon suivante :
"N :" chaine de caractères representant le module n en base 10.
"E :" chaine de caractères representant l'exposant e en base 10.

2.1.4 chargement de clé

Dans notre application, il est possible de charger les clés directement à partir d'un bouton. Ce bouton ouvre une boite de dialogue qui permet de selectionner le fichier de clé.

Pour le chiffrement

Dans la boite de dialogue de chiffrement, le bouton permet uniquement de charger les fichiers de clé publique (*.public).

Pour le déchiffrement

Dans la boite de dialogue de déchiffrement, le bouton permet uniquement de charger les fichiers de clé privée (*.private).

Pour la signature

Dans la boite de dialogue de signature, le bouton permet uniquement de charger les fichiers de clé privée (*.private).

Pour la vérification de provenance

Dans la boite de dialogue de vérification de la provenance d'un message, le bouton permet uniquement de charger les fichiers de clé publique (*.public).

Chapitre 3

Les jeux de tests