**LYCEE LA SALLE PASSY BUZENVAL**



***Projet :***

***CRYPTOMESSENGER***



**PAUL NADAL**

**Terminale SL**

**Option Sciences de l’Ingénieur**

**Spécialité Informatique et Sciences du Numérique**

**Le 7 juin 2018**

**Groupe de travail :**

*Paul Nadal*

*Julien Marzal-Hilaire*

*Donatien Philippo*

**Sommaire :**

[Présentation du Projet : 3](#_Toc515818098)

[1. Chiffres et actualités 3](#_Toc515818099)

[2. Solutions mises en place 3](#_Toc515818100)

[3. Notre solution : CryptoMessenger 3](#_Toc515818101)

[Cahier des charges fonctionnel : 4](#_Toc515818102)

[1. Bête à cornes 4](#_Toc515818103)

[2. Diagramme utilisateur 4](#_Toc515818104)

[Répartition des tâches : 5](#_Toc515818105)

[Fonctionnement du programme : 5](#_Toc515818106)

[1. Navigation dans l’interface 5](#_Toc515818107)

[2. Fonctionnement du logiciel 6](#_Toc515818108)

[Explication de la méthode de cryptage DES : 7](#_Toc515818109)

[1. Création des 16 clés intermédiaires 7](#_Toc515818110)

[2. Cryptage du message 8](#_Toc515818111)

[3. Histoire du DES 9](#_Toc515818112)

[Problèmes rencontrés et améliorations futures : 10](#_Toc515818113)

[1. Problèmes rencontrés 10](#_Toc515818114)

[2. Améliorations futures 10](#_Toc515818115)

[Annexes 11](#_Toc515818116)

[1. Tables Sn 11](#_Toc515818117)

[2. Code du Cryptage 13](#_Toc515818118)

[3. Code de l’interface client 24](#_Toc515818119)

[4. Code de l’interface client 30](#_Toc515818120)

Présentation du Projet :

1. Chiffres et actualités

Plus de 8 entreprises françaises sur 10 ont été visées par une cyberattaque en 2015.

Cela signifie que 80% des entreprises françaises se font voler des données importantes et confidentielles, ce qui peut entraîner leur faillite.

Cela est dû vraisemblablement à un manque de sécurisation des données.

Beaucoup d’informations sont traitées par de nombreux intermédiaires et sont ainsi insuffisamment sécurisées.

Des logiciels créés par des hackers permettent de récupérer et lire des données envoyées par mail ou via les réseaux sociaux par exemple.

1. Solutions mises en place

Des solutions pour sécuriser les données sont d’ores et déjà disponibles. Les mails sont cryptés, des préventions sur les sites frauduleux sont mises en place partout en France et dans le monde. En effet, des logiciels d’antivirus gratuits ou payants sont disponibles sur le marché afin de protéger les ordinateurs des malwares et permettent déjà de corriger les failles permettant aux hackers de pirater les données.

Cependant, des failles persistent toujours et les cyberpirates continuent à chercher des solutions afin de « cracker » les programmes de cryptage et de sécurisation des données personnelles.





1. Notre solution : CryptoMessenger

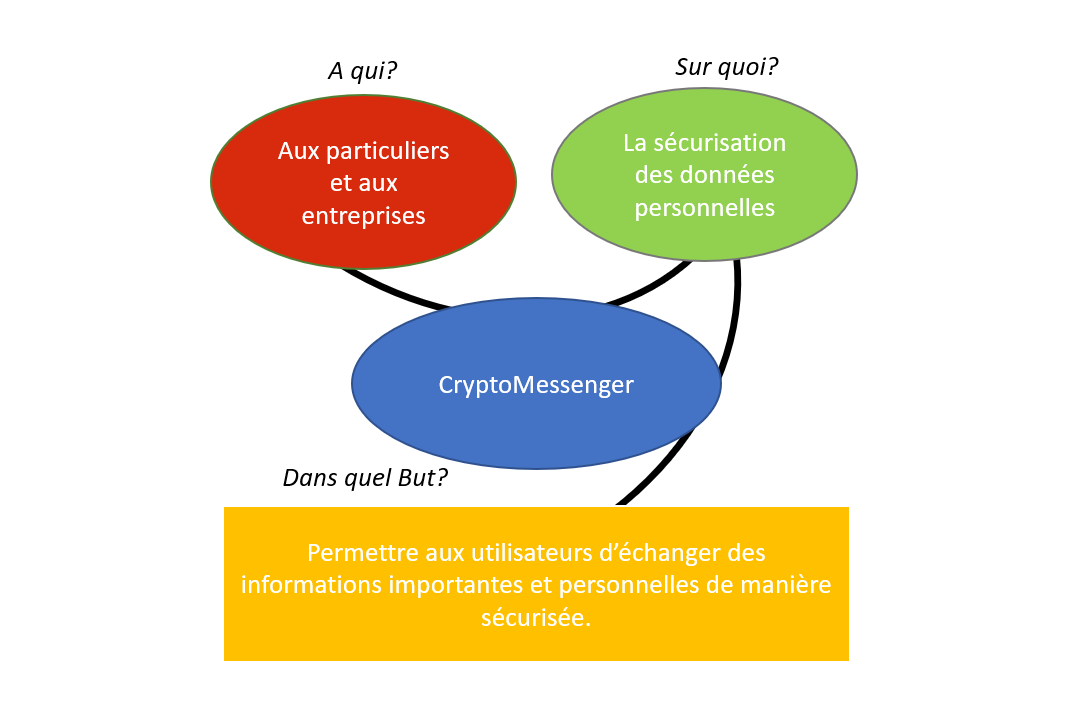
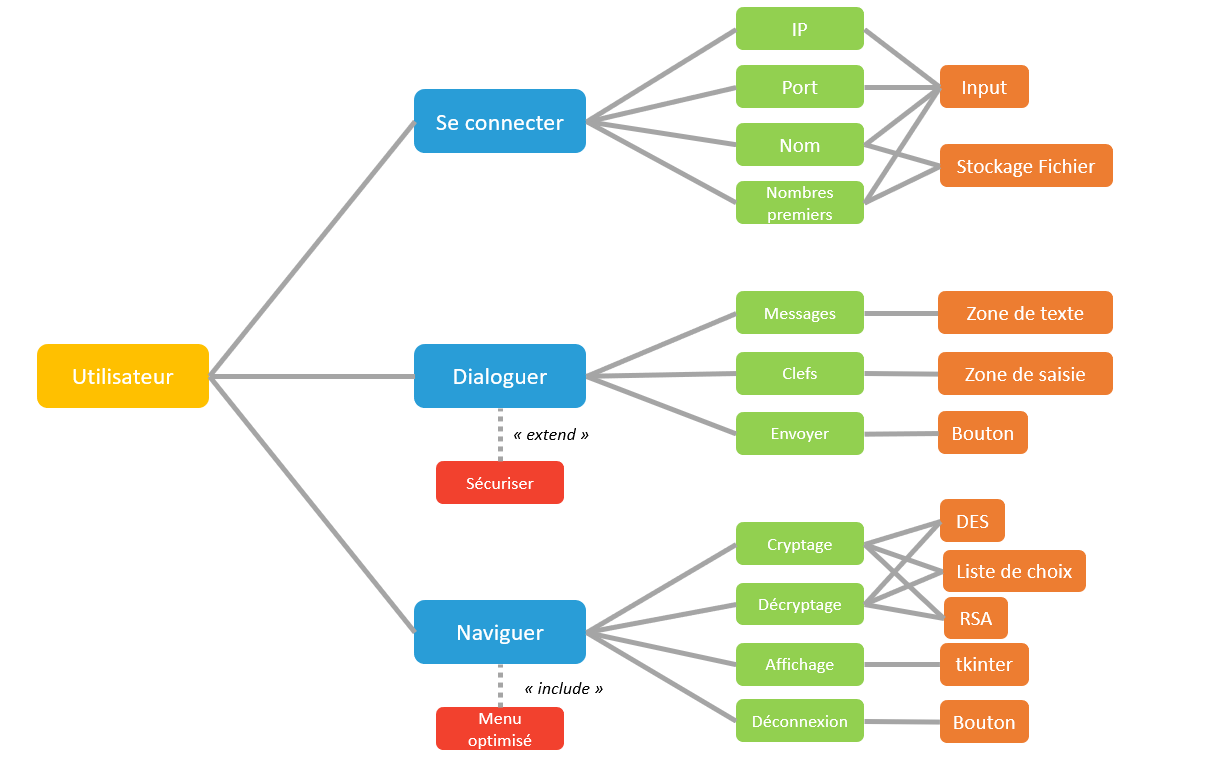
Afin de remédier à ces problèmes de cyberattaque, nous avons conçu un logiciel de sécurisation de transmission des messages.

CryptoMessenger est un programme de communication sécurisée (messagerie instantanée) permettant d’échanger avec un groupe de personnes connectées sur un même serveur. Les utilisateurs cryptent les messages avant de les envoyer aux autres.

Les méthodes de cryptage utilisées sont le RSA et le DES. Elles permettent de crypter les messages et clefs utilisées afin que seules les personnes connectées puissent les décrypter.

Ce programme permet donc de discuter avec d’autres personnes sans se soucier de la sécurité des données partagées. Les méthodes de cryptage utilisées dans notre programme ne sont pas infaillibles mais assurent néanmoins un maximum de sécurité au vu de la complexité à « cracker » ces cryptages.

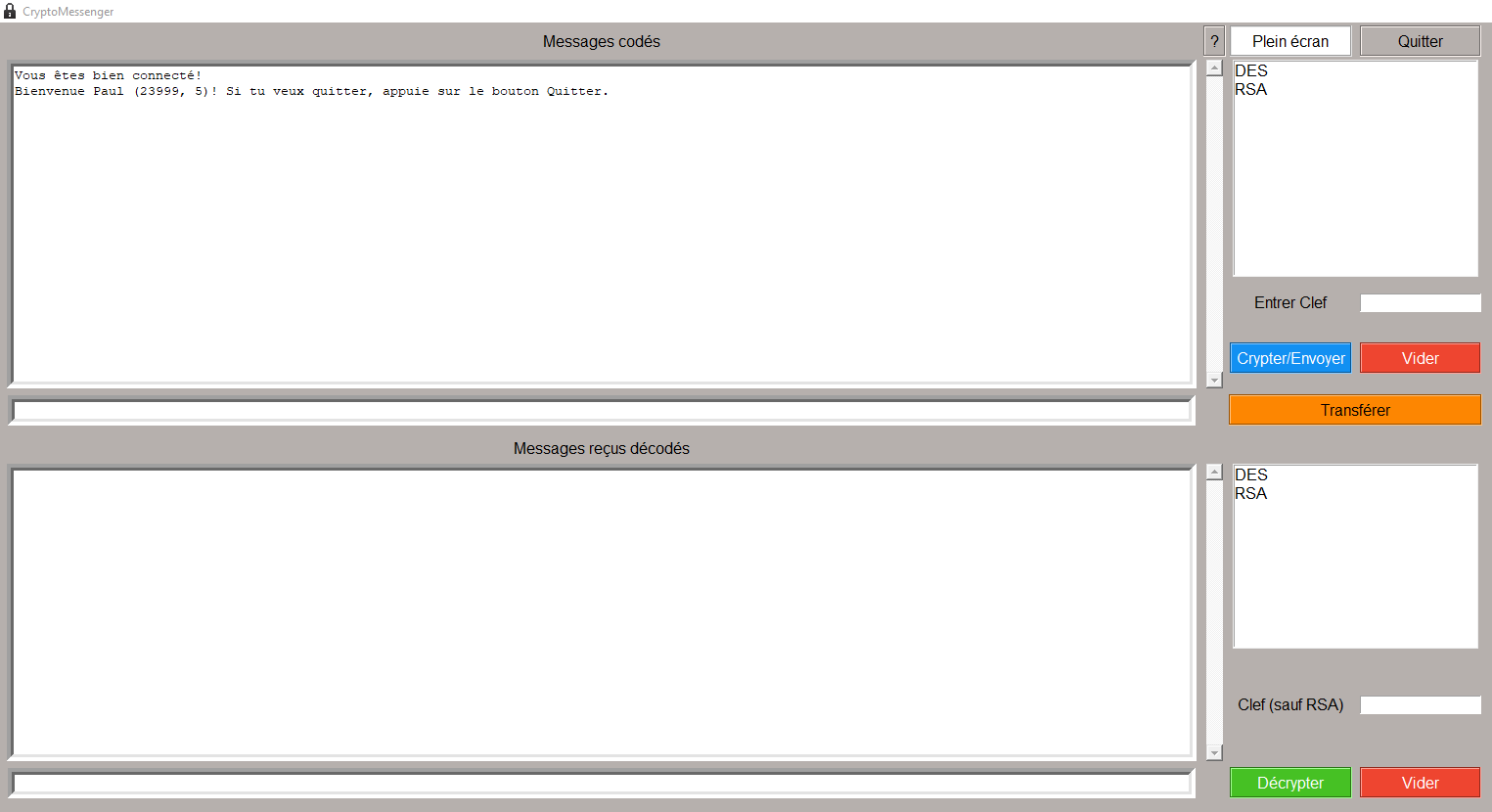
Cahier des charges fonctionnel :

1. Bête à cornes
2. Diagramme utilisateur

Répartition des tâches :

|  |  |  |
| --- | --- | --- |
| Paul | Julien | Donatien |
| Cryptage DES | Interface Tkinter | Cryptage RSA |
| Client + Serveur | | |

Fonctionnement du programme :

1. Navigation dans l’interface

Permet de crypter et envoyer le message tapé dans la zone de saisie dédiée avec la méthode de cryptage sélectionnée.

Crypter/Envoyer

Permet de décrypter le message codé tapé dans la zone de saisie dédiée avec la méthode de cryptage sélectionnée.

Décrypter

Permet de vider la zone de réception des messages, la zone de saisie des messages et de la clef.

Vider

Permet de transférer le dernier message reçu dans la zone de réception des messages codés.

Transférer

?

Permet d’afficher la fenêtre d’aide en cas de besoin.

1. Fonctionnement du logiciel

***Comment se connecter ?***

Un des utilisateurs du groupe doit lancer le programme Serveur.py à l’aide de Python.

Lancez le programme Cryptomessenger.py à l’aide de Python ou Edupython. Entrez l’adresse IP de l’ordinateur qui héberge le serveur dans la zone de texte dédiée ou sélectionnez-en une précédemment utilisée dans l’historique des IP. Entrez votre nom, 2 nombres premiers différents entre 100 et 1000, puis le port utilisé (33 000 par défaut).

***Comment envoyer un message crypté ?***

Envoyer un message crypté se fait en 2 étapes :

Vous pouvez pour commencer, envoyer en clair le nom de la personne à qui est destiné le message.

* Envoi du message : Tapez le message à envoyer dans la barre de texte dans la zone "Messages codés".

Sélectionnez ensuite DES dans la liste des méthodes de cryptage.

Tapez votre clef dans la zone définie, puis cliquez sur le bouton Crypter/Envoyer.

Le message est envoyé en crypté (Hexadécimal) et s’affiche dans la zone de réception des messages codés.

* Envoi de la clef : Tapez la clef que vous avez précédemment utilisée dans la barre de texte dans la zone Messages codés. Sélectionnez ensuite RSA dans la liste des méthodes de cryptage. Tapez la clef publique de la personne à qui s’adresse le message (ne pas oublier les parenthèses), que vous trouvez à côté de son nom, puis cliquez sur le bouton Crypter/Envoyer. La clef est envoyée en crypté (Chiffres) et s’affiche dans la zone de réception des Messages codés.

***Comment décrypter un message reçu ?***

Décrypter un message se fait en 2 étapes.

* Décryptage de la clef : Appuyez sur Transférer pour déplacer le dernier message reçu (clef cryptée) dans la barre de texte de la zone "Messages décodés". Vous pouvez aussi le faire manuellement pour des messages et clefs reçus précédemment. Sélectionnez ensuite RSA dans la liste des méthodes de décryptage, puis cliquez sur le bouton Décrypter. La clef claire s’affiche dans la zone de réception des Messages décodés et dans la zone définie pour la clef.
* Décryptage du message : Copiez/Collez le message crypté dans la barre de texte dans la zone Messages décodés. Sélectionnez ensuite DES dans la liste des méthodes de décryptage, puis cliquez sur le bouton Décrypter. La clef est déjà entrée dans la zone définie précédemment. Le message clair s’affiche dans la zone de réception des Messages décodés.

Explication de la méthode de cryptage DES :

1. Création des 16 clés intermédiaires

On choisit une clé de 8 caractères maximum. On la convertit en binaire (64 bits).

On permute l’ordre des bits de la clef et on divise cette permutation en 2 parties (**Cn** et **Dn**) selon les tableaux suivants :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **63** | **55** | **47** | **39** | **31** | **23** | **15** |
| **7** | **62** | **54** | **46** | **38** | **30** | **22** |
| **14** | **6** | **61** | **53** | **45** | **37** | **29** |
| **21** | **13** | **5** | **28** | **20** | **12** | **4** |

**K+ = C0 + D0 :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **57** | **49** | **41** | **33** | **25** | **17** | **9** |
| **1** | **58** | **50** | **42** | **34** | **26** | **18** |
| **10** | **2** | **59** | **51** | **43** | **35** | **27** |
| **19** | **11** | **3** | **60** | **52** | **44** | **36** |

On effectue ensuite une boucle de 16 itérations :

On commence par placer le premier ou les deux premiers caractères à la fin selon le numéro de l’itération en cours.

|  |  |
| --- | --- |
| 9 | 1 |
| 10 | 2 |
| 11 | 2 |
| 12 | 2 |
| 13 | 2 |
| 14 | 2 |
| 15 | 2 |
| 16 | 1 |

**Rotations de Cn et Dn par la droite :**

|  |  |
| --- | --- |
| N° itération | Nb de permutations |
| 1 | 1 |
| 2 | 1 |
| 3 | 2 |
| 4 | 2 |
| 5 | 2 |
| 6 | 2 |
| 7 | 2 |
| 8 | 2 |

On place bout à bout les deux parties **Cn** et **Dn** avant de permuter à nouveau selon le tableau suivant.

**Permutation PC2 pour chaque paire CnDn (avec 1≤ n≤ 16) :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 14 | 17 | 11 | 24 | 1 | 5 |
| 3 | 28 | 15 | 6 | 21 | 10 |
| 23 | 19 | 12 | 4 | 26 | 8 |
| 16 | 7 | 27 | 20 | 13 | 2 |
| 41 | 52 | 31 | 37 | 47 | 55 |
| 30 | 40 | 51 | 45 | 33 | 48 |
| 44 | 49 | 39 | 56 | 34 | 53 |
| 46 | 42 | 50 | 36 | 29 | 32 |

A la fin de la boucle, on obtient les 16 clefs intermédiaires (48 bits) qui seront utilisées pour crypter le message.

Pour décrypter, on inverse seulement l’ordre des clefs.

1. Cryptage du message

On choisit un message de la taille souhaité. On divise ce message en groupe de 8 caractères qu’on cryptera tour à tour. On convertit chaque groupe en binaire (64 bits).

On procède à une permutation initiale comme pour la clef qu’on divise en 2 parties (**Ln** et **Rn**) selon les tableaux suivants :

**M+ = L0 + R0 :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **57** | **49** | **41** | **33** | **25** | **17** | **9** |
| **1** | **58** | **50** | **42** | **34** | **26** | **18** |
| **10** | **2** | **59** | **51** | **43** | **35** | **27** |
| **19** | **11** | **3** | **60** | **52** | **44** | **36** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **63** | **55** | **47** | **39** | **31** | **23** | **15** |
| **7** | **62** | **54** | **46** | **38** | **30** | **22** |
| **14** | **6** | **61** | **53** | **45** | **37** | **29** |
| **21** | **13** | **5** | **28** | **20** | **12** | **4** |

On procède aussi à 16 itérations liées à celles de la clef.

**Procédure x16 :**

**Ln** = **Rn-1**

**Rn** = **Ln-1** + *f* (**Rn-1**, **Kn**)

On expand tout d’abord **Rn-1** à 48 bits.

**Expansion de Rn:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 32 | 1 | 2 | 3 | 4 | 5 |
| 4 | 5 | 6 | 7 | 8 | 9 |
| 8 | 9 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 30 | 31 | 32 | 1 |

On calcule **Kn** + E(**Rn-1**) avec XOR (OU exclusif) avec **Kn**, la clé de la nième itération.

**Tables Sn :**

On divise le résultat obtenu en 8 groupes de 6 bits et on procède comme indiqué ci-dessous :

**Kn** + E(**Rn-1**) =B1B2B3B4B5B6B7B8

On calcule: S1(B1)S2(B2)S3(B3)S4(B4)S5(B5)S6(B6)S7(B7)S8(B8)

Le 1er et le dernier bit de chaque groupe de 6 bits représentent en base 2 la ligne entre 0 et 3.

Les 4 bits du milieu représentent en base 2 la colonne entre 0 et 15.

Avec ces coordonnées, on lit le numéro (entre 0 et 15) dans la table Sn, ce qui donne 4 bits en binaire. Les tables Sn sont définies dans les annexes.

On permute à nouveau le résultat de S1(B1)S2(B2)S3(B3)S4(B4)S5(B5)S6(B6)S7(B7)S8(B8) pour obtenir *f* (**Rn-1**, **Kn**) :

**Permutation P:**

*f* (**Rn-1**, **Kn**) = P(S1(B1)S2(B2)S3(B3)S4(B4)S5(B5)S6(B6)S7(B7)S8(B8))

|  |  |  |  |
| --- | --- | --- | --- |
| 16 | 7 | 20 | 21 |
| 29 | 12 | 28 | 17 |
| 1 | 15 | 23 | 26 |
| 5 | 18 | 31 | 10 |
| 2 | 8 | 24 | 14 |
| 32 | 27 | 3 | 9 |
| 19 | 13 | 30 | 6 |
| 22 | 11 | 4 | 25 |

On procède à nouveau à un XOR entre **Ln-1** et *f* (**Rn-1**, **Kn**) pour obtenir **Rn**.

On répète l’opération 16 fois pour obtenir **L16** et **R16**.

On inverse l’ordre de L et R pour obtenir : M16 = **R16L16**.

On permute une dernière fois pour obtenir le message codé final.

**Permutation finale :**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 40 | 8 | 48 | 16 | 56 | 24 | 64 | 32 |
| 39 | 7 | 47 | 15 | 55 | 23 | 63 | 31 |
| 38 | 6 | 46 | 14 | 54 | 22 | 62 | 30 |
| 37 | 5 | 45 | 13 | 53 | 21 | 61 | 29 |
| 36 | 4 | 44 | 12 | 52 | 20 | 60 | 28 |
| 35 | 3 | 43 | 11 | 51 | 19 | 59 | 27 |
| 34 | 2 | 42 | 10 | 50 | 18 | 58 | 26 |
| 33 | 1 | 41 | 9 | 49 | 17 | 57 | 25 |

On obtient donc le message crypté. Pour décrypter, il suffit d’inverser les clefs.

1. Histoire du DES

En 1970, l’armée américaine avait besoin d’une nouvelle méthode de cryptage et lança donc un concours. IBM l’emporta avec son algorithme nommé Lucifer. Après quelques améliorations, ce cryptage fut approuvé en 1977 sous le nom de DES (*Data Encryption Standard*). Le DES demande normalement une clef et un message de 8 caractères chacun. Nous avons voulu dans notre programme améliorer cela en découpant un message plus long, afin de coder des groupes de 8 caractères et pouvoir ainsi communiquer plus facilement.

La NSA a eu du mal à « cracker » ce code sachant qu’il existe plus de 7,2 **.** 1016 possibilités de clef. Cela fait du DES une méthode de cryptage très sécurisée et très difficilement attaquable.

Problèmes rencontrés et améliorations futures :

1. Problèmes rencontrés

Ce qui nous a posé le plus de souci dans le programme a été la gestion du serveur/client. L’envoi, la distribution et la réception des messages étaient en effet compliqués à gérer en même temps.

Au début, nous voulions que chaque utilisateur soit à la fois serveur et client mais ce n’était pas possible car le programme ne pouvait pas gérer les deux fonctions à la fois.

Nous avons finalement opté pour un serveur lancé à partir d’un deuxième programme séparément. Ainsi, un seul des utilisateurs doit lancé le serveur et tous les utilisateurs y compris lui, peuvent s’y connecter et devenir client afin d’envoyer et recevoir des messages.

Le second problème que nous avons rencontré est la grande taille de notre code. Nous avons donc choisi de séparer l’interface client des méthodes de cryptage en faisant de l’un des programmes une librairie utilisable par l’autre programme.

1. Améliorations futures

Notre programme fonctionne correctement à ce stade de son développement mais quelques améliorations d’ergonomie sont néanmoins envisageables :

* Listes des utilisateurs connectés visibles dans notre programme afin de pouvoir sélectionner plus facilement la clef publique du destinataire.
* Ecran d’accueil permettant de rentrer facilement l’IP, le port et permettant de changer le nom et les nombres premiers du RSA si l’utilisateur le souhaite.
* Ajout d’une liste des adresses IP déjà utilisées afin de se connecter plus facilement.
* Gestion des erreurs lors de la connexion et de l’utilisation du programme.
* Convertir le programme en .exe afin de pouvoir distribuer notre logiciel plus facilement.

Nous avons déjà mis en place certaines de ces améliorations dans notre programme car nous participons au concours d’ISN : *Labex Digicosme*.

Annexes

1. Tables Sn

S1 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 14 | 4 | 13 | 1 | 2 | 15 | 11 | 8 | 3 | 10 | 6 | 12 | 5 | 9 | 0 | 7 |
| **1** | 0 | 15 | 7 | 4 | 14 | 2 | 13 | 1 | 10 | 6 | 12 | 11 | 9 | 5 | 3 | 8 |
| **2** | 4 | 1 | 14 | 8 | 13 | 6 | 2 | 11 | 15 | 12 | 9 | 7 | 3 | 10 | 5 | 0 |
| **3** | 15 | 12 | 8 | 2 | 4 | 9 | 1 | 7 | 5 | 11 | 3 | 14 | 10 | 0 | 6 | 13 |

S2 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 15 | 1 | 8 | 14 | 6 | 11 | 3 | 4 | 9 | 7 | 2 | 13 | 12 | 0 | 5 | 10 |
| **3** | 3 | 13 | 4 | 7 | 15 | 2 | 8 | 14 | 12 | 0 | 1 | 10 | 6 | 9 | 11 | 5 |
| **2** | 0 | 14 | 7 | 11 | 10 | 4 | 13 | 1 | 5 | 8 | 12 | 6 | 9 | 3 | 2 | 15 |
| **3** | 13 | 8 | 10 | 1 | 3 | 15 | 4 | 2 | 11 | 6 | 7 | 12 | 0 | 5 | 14 | 9 |

S3 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 10 | 0 | 9 | 14 | 6 | 3 | 15 | 5 | 1 | 13 | 12 | 7 | 11 | 4 | 2 | 8 |
| **1** | 13 | 7 | 0 | 9 | 3 | 4 | 6 | 10 | 2 | 8 | 5 | 14 | 12 | 11 | 15 | 1 |
| **2** | 13 | 6 | 4 | 9 | 8 | 15 | 3 | 0 | 11 | 1 | 2 | 12 | 5 | 10 | 14 | 7 |
| **3** | 1 | 10 | 13 | 0 | 6 | 9 | 8 | 7 | 4 | 15 | 14 | 3 | 11 | 5 | 2 | 12 |

S4 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 7 | 13 | 14 | 3 | 0 | 6 | 9 | 10 | 1 | 2 | 8 | 5 | 11 | 12 | 4 | 15 |
| **1** | 13 | 8 | 11 | 5 | 6 | 15 | 0 | 3 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |
| **2** | 10 | 6 | 9 | 0 | 12 | 11 | 7 | 13 | 15 | 1 | 3 | 14 | 5 | 2 | 8 | 4 |
| **3** | 3 | 15 | 0 | 6 | 10 | 1 | 13 | 8 | 9 | 4 | 5 | 11 | 12 | 7 | 2 | 14 |

S5 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 2 | 12 | 4 | 1 | 7 | 10 | 11 | 6 | 8 | 5 | 3 | 15 | 13 | 0 | 14 | 9 |
| **1** | 14 | 11 | 2 | 12 | 4 | 7 | 13 | 1 | 5 | 0 | 15 | 10 | 3 | 9 | 8 | 6 |
| **2** | 4 | 2 | 1 | 11 | 10 | 13 | 7 | 8 | 15 | 9 | 12 | 5 | 6 | 3 | 0 | 14 |
| **3** | 11 | 8 | 12 | 7 | 1 | 14 | 2 | 13 | 6 | 15 | 0 | 9 | 10 | 4 | 5 | 3 |

S6 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 12 | 1 | 10 | 15 | 9 | 2 | 6 | 8 | 0 | 13 | 3 | 4 | 14 | 7 | 5 | 11 |
| **1** | 10 | 15 | 4 | 2 | 7 | 12 | 9 | 5 | 6 | 1 | 13 | 14 | 0 | 11 | 3 | 8 |
| **2** | 9 | 14 | 15 | 5 | 2 | 8 | 12 | 3 | 7 | 0 | 4 | 10 | 1 | 13 | 11 | 6 |
| **3** | 4 | 3 | 2 | 12 | 9 | 5 | 15 | 10 | 11 | 14 | 1 | 7 | 6 | 0 | 8 | 13 |

S7 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 4 | 11 | 2 | 14 | 15 | 0 | 8 | 13 | 3 | 12 | 9 | 7 | 5 | 10 | 6 | 1 |
| **1** | 13 | 0 | 11 | 7 | 4 | 9 | 1 | 10 | 14 | 3 | 5 | 12 | 2 | 15 | 8 | 6 |
| **2** | 1 | 4 | 11 | 13 | 12 | 3 | 7 | 14 | 10 | 15 | 6 | 8 | 0 | 5 | 9 | 2 |
| **3** | 6 | 11 | 13 | 8 | 1 | 4 | 10 | 7 | 9 | 5 | 0 | 15 | 14 | 2 | 3 | 12 |

S8 :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **0** | 13 | 2 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| **1** | 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12 | 5 | 6 | 11 | 0 | 14 | 9 | 2 |
| **2** | 7 | 11 | 4 | 1 | 9 | 12 | 14 | 2 | 0 | 6 | 10 | 13 | 15 | 3 | 5 | 8 |
| **3** | 2 | 1 | 14 | 7 | 4 | 10 | 8 | 13 | 15 | 12 | 9 | 0 | 3 | 5 | 6 | 11 |

1. Code du Cryptage
2. Code de l’interface client
3. Code du serveur