GYMNASE DE BEAULIEU
TRAVAIL DE MATURITE 2018

Cryptographie et Programmation

M. Leroy-Beaulieu

Calum Quinn

Villars-Tiercelin, le 2 décembre 2018

Résumé

Le but de mon travail de maturité était d'approfondir mes connaissances en cryptographie et d'apprendre à programmer.

Dans ce travail j'explique:

- le principe du cryptage;
- la création, le fonctionnement et le codage de mon algorithme personnel;
- le fonctionnement et le codage de l'algorithme RSA;
- ma procédure pour ce travail.

Tout d'abord, je parle brièvement des utilités du cryptage, d'il y a longtemps et d'aujourd'hui.

Ensuite je décris comment je suis venu à créer mon algorithme et j'explique en détail toutes les étapes du fonctionnement de cet algorithme en donnant quelques exemples. Après avoir expliqué comment fonctionne cet algorithme, je parle de comment je suis parvenu à coder une implémentation de cet algorithme dans un programme informatique.

J'explique ensuite comment fonctionne l'algorithme RSA et la manière de laquelle j'ai programmé son implémentation.

Finalement je décris la procédure de ce travail et mes sentiments envers celui-ci.

Table des matières

Cryptographie et Programmation

1 Introduction

1.1 Mon choix

Le choix du thème et même de l'idée de ce travail n'était pas d'une grande difficulté. Dès qu'on a commencé à me parler de travail de maturité, je savais que je voulais faire le mien dans le domaine de la cryptographie. En ce qui concerne le concept précis, j'ai eu quelques idées telles qu'une machine physique pour crypter de façon mécanique des messages. Mais finalement, j'ai décidé de créer mon propre algorithme de cryptage car c'est l'idée qui me passionnait le plus.

1.2 But

Le but de mon travail de maturité ou TM, était en première partie d'approfondir mes connaissances en cryptographie avec plusieurs algorithmes de cryptage, l'un que j'ai inventé et que j'ai appelé algorithme Travail de Cryptographie et Programmation ou algorithme TCP et l'autre qui est nommé d'après ces inventeurs, Ron Rivest, Adi Shamir, et Leonard Adleman donc algorithme RSA. La deuxième partie était d'apprendre à programmer à un niveau suffisamment élevé pour programmer des systèmes basiques et l'algorithme TCP. Finalement, le but d'un travail de maturité est, en général de nous apprendre à produire un travail conséquent en s'investissant du début à la fin.

1.3 Principe du cryptage

Le principe de crypter un texte existe depuis des milliers d'années et sert à cacher le contenu d'un message pour que seulement la ou les personnes qui possèdent les outils nécessaires puissent le décrypter et donc le comprendre. Il y a bien sûr énormément de manières de crypter qui ont été inventées au cours des années avec l'avancement technologique. Il y en a qui sont simples à comprendre et il y en a aussi qui sont compliquées. Les plus compliquées d'entre elles sont utilisées par exemple pour la protection de données informatiques personnelles et sensibles telles que les coordonnées bancaires.

2 L'algorithme TCP

2.1 Création

Pour ce TM, un algorithme de cryptage qui n'avait pas été utilisé précédemment a été créé. Il provient d'inspiration d'autres algorithmes comme par exemple celui d'Enigma qui était le système utilisé par les Allemands pendant la Seconde Guerre mondiale [2] [3]. Après plusieurs essais, l'algorithme TCP a été créé et il était assez sécurisé pour ne pas pouvoir être décrypté par une personne qui n'est pas spécialisée dans ce domaine. Sa sécurité est expliquée dans la partie suivante.
Le fonctionnement de cet algorithme était testé avec le cryptage et le décryptage de messages variés.

2.2 Fonctionnement

L'objectif de cette partie est d'expliquer le fonctionnement de l'algorithme TCP d'une manière compréhensible.
Le principal but de cet algorithme est de transformer un texte quelconque, que l'on introduit, en quelque chose d'illisible si le lecteur ne possède pas les outils nécessaires. Le plus important de ces outils est la clé de chiffrage et de déchiffrage, qui est composée de cinq nombres (a, b, c, d, e) compris dans l'intervalle $[2; 11]$, ces cinq nombres sont choisis au moment de crypter le message et sont réutilisés lors du décryptage. La manière que l'algorithme emploie pour rendre un message illisible, est de remplacer chaque lettre par une autre. Ceci peut être fait de deux manières différentes, de façon monoalphabétique ou polyalphabétique. TCP est un système polyalphabétique car cela est bien plus sécurisé que les systèmes monoalphabétiques. La raison pour laquelle les systèmes de ce type sont plus sécurisés est que chaque lettre est cryptée avec un alphabet différent, donc si on parvient à décrypter un message, un autre message ne sera pas décryptable de la même façon.
Pour définir les différents alphabets qui seront utilisés pour remplacer les lettres du message, l'algorithme déplace certaines lettres de l'alphabet, un certain nombre de positions, relatives aux autres lettres. Ces déplacements sont effectués selon des couples de deux nombres $(m; n)$. Le premier de ces nombres nous indique que nous allons déplacer chaque $m^{ème}$ lettre de l'alphabet. Le deuxième nombre n représente le nombre de places à droite dont nous allons déplacer les lettres choisies à l'étape précédente.
Ces couples sont définis par la clé de cryptage, le premier nombre a de la clé de cryptage nous indique combien de nombres différents il pourrait y avoir dans les couples. Si $a = 4$, cela veut dire que nous pourrions avoir les

4 premiers nombres de l'intervalle $[2; 11]$ dans nos couples. C'est-à-dire les nombres 2, 3, 4 et 5. Il y a une spécialité si le premier nombre est 11, dans cette situation il est possible d'avoir les nombres entre 2 et 12.
Le deuxième nombre b de la clé crée une liste avec les $b^{ème}$ nombres de la liste créée avec a, donc l'intervalle $[2; a + 1]$ ($a + 1$ car l'intervalle commence à 2 et non pas 1). Reprenons notre exemple avec $a = 4$, définissons que $b = 3$, c'est-à-dire que la liste des $b^{ème}$ nombres est de la forme $b1, b2, b3, ...$ ou 4, 3, 2, ... Voici une image pour représenter la situation:

Figure 1: Choix des $b^{ème}$ nombres
A la première ligne la liste des $a^{ème}$ premiers nombres de l'intervalle $[2; 11]$, à la deuxième ligne un compteur de 1 à b qui se répète infiniment et surligné chaque $b^{ème}$ nombre de la liste fait avec a

En ce qui concerne le troisième nombre c, le principe est identique à celui du deuxième nombre. Pour l'exemple, supposons que $c = 5$; ceci produirait une deuxième liste de même sorte que celle avec les $b^{ème}$ nombres mais cette fois ce serait les nombres $c1, c2, c3, ...$ ou 2, 3, 4, ...
La prochaine étape est de mettre ensemble ces deux listes de sorte à créer nos couples. Ceci est fait en mettant les nombres $b1$ et $c1$ ensemble $(4; 2)$, les nombres $b2$ et $c2$ ensemble $(3; 3)$ et ainsi de suite pour avoir une liste *couples* infinie (dans ce cas, la liste se répète après quatre couples).
Les deux derniers nombres de la clé sont seulement utiles si nous faisons plusieurs changements au même alphabet avant d'en extraire la lettre souhaitée, et c'est ce que fait l'algorithme TCP. Donc le quatrième nombre d de cette clé indique combien de changements nous ferons à l'alphabet pour chaque lettre. Dans notre exemple prenons $d = 3$. Ceci voudrait dire que nous ferions les 3 premiers changements de notre liste *couples*, c'est-à-dire $(4; 2)$, $(3; 3)$ et $(2; 4)$.

Finalement, le cinquième nombre e concerne quels changements nous faisons à l'alphabet avant de choisir la lettre voulue. Pour la première lettre du message que nous voulons crypter, nous allons faire les d premiers changements de notre liste *couples*. Par contre pour la prochaine lettre nous allons reprendre l'alphabet normal et lui faire aussi d changements. Les changements que nous allons faire sont les d qui viennent après le nombre e de couples. La lettre d'après sera choisie avec les d couples qui viennent après $2 * e$ couples. Dû à la complexité de cette explication, quelques images ont été rajoutées.

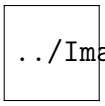

Figure 2: Couples utilisés
A la première ligne le rectangle bleu encadre les d premiers changements qui seront faits à l'alphabet pour la première lettre à crypter. A la deuxième ligne le rectangle vert encadre les e premiers changements donc le rectangle bleu qui se trouve à droite encadre les d prochains changements qui seront faits à l'alphabet pour la deuxième lettre du message et à la dernière ligne les deux encadrés vert encadrent les $2 * e$ premiers changements et le rectangle bleu encadre les d changements qui seront faits à l'alphabet pour la troisième lettre à crypter

Les images qui suivent illustrent les déplacements faits dans l'alphabet pour un changement $(4; 2)$.

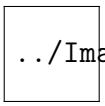

Figure 3: Lettres à déplacer
A la première ligne se trouve l'alphabet normal, à la deuxième ligne se trouve un compteur allant de 1 à m du couple $(m; n)$ ici $(4; 2)$ commençant par m et qui se répètent jusqu'au bout de l'alphabet et en surligné c'est chaque $m^{ème}$ lettre de l'alphabet

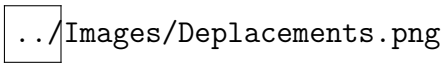

Figure 4: Déplacements effectués
La ligne du milieu représente l'alphabet normal, les flèches montrent quelles lettres sont déplacées et où. Elles sont insérées dans les cases vides de la deuxième ligne et à la troisième ligne se trouvent des compteurs des $n^{ème}$ (dans ce cas 2) lettres se trouvant après les lettres déplacées qui montrent où les lettres déplacées sont insérées

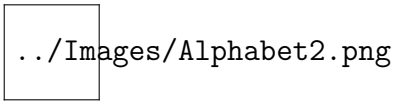

Figure 5: Alphabet après changements
Cette ligne montre l'alphabet après un changement $(4; 2)$

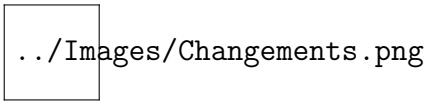

Figure 6: Exemples d'alphabets changés
Le tableau d'en haut montre l'alphabet normal avec en dessous l'alphabet après le changement $(4; 2)$ et le tableau d'en bas contient l'alphabet normal et dessous l'alphabet après un changement $(2; 3)$

Après ces déplacements effectués, nous avons encore un alphabet à 26 lettres, donc pour rendre le message illisible nous allons prendre la lettre étant à la même position que celle que nous voulons remplacer et la remplacer par la lettre à la même position dans l'alphabet avec les changements.
Par exemple si nous voulions remplacer la lettre J dans un texte, vu que la lettre J est à la 10ème position de l'alphabet, nous allons la remplacer par la lettre qui sera à la 10ème position de l'alphabet avec changements. Dans cet exemple nous remplacerions la lettre J par la lettre G.

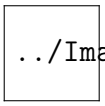

Figure 7: Changement d'une lettre
Le premier tableau contient les 14 premières lettres de l'alphabet normal avec la $10^{ème}$ surlignée et le deuxième tableau contient les 14 premières lettres de l'alphabet après un changement $(2; 3)$ avec aussi la $10^{ème}$ lettre surlignée, ceci démontre que si notre message contenait la lettre J et que nous avions fait le changement $(2; 3)$, nous l'aurions remplacée par un G

Étapes:

1. Choisir cinq nombres dans l'intervalle [2;11] (on peut prendre plusieurs fois le même). (EX: 5, 6, 3, 5, 10)

2. La clé de cryptage est la concaténation des 5 nombres choisis précédemment. (EX: 563510)

3. A l'aide de cette clé nous trouvons jusqu'à 132^{11} ou $2.12*10^{23}$ alphabets différents avec les lettres "mélangées".

4. Chaque lettre du message à crypter correspond à un alphabet différent (si la longueur du message excède le nombre d'alphabets disponibles, plusieurs lettres correspondront au même alphabet).

5. Pour chaque lettre du message à crypter, trouver sa position dans l'alphabet (EX: e $\Rightarrow$ 5).

6. Réécrire le message en prenant à chaque fois la lettre équivalente dans l'alphabet correspondant.

7. Le décryptage d'un message consiste à faire les trois premières étapes à l'aide de la clé de cryptage/décryptage de la même façon qu'avant et ensuite de chercher les lettres dans les alphabets avec changements pour ensuite reprendre les lettres à la même position de l'alphabet normal. Cette dernière partie revient à faire l'inverse des étapes 5 et 6.

Exemple

Voici une petite démonstration du fonctionnement de l'algorithme TCP avec le déroulement complet de toutes les étapes expliquées au point précédent.

Cette étape reprend les mêmes nombres que l'explication d'avant.

Cryptage

1. Clé de cryptage: 43532
2. Liste des b: 4, 3, 2, 5,... et liste des c: 2, 3, 4, 5,...
3. Liste *couples*: $(4;2), (3;3), (2;4), (5;5), ...$
4. Message à crypter: salut
5. Changement $(4;2)$, l'alphabet normal devient:
   y b c a d f g e h j k i l n o m p r s q t v w u x z
   Changement $(3;3)$:
   b x c y d f a e h g k i j n o l p r m q t s w u v z
   Changement $(2;4)$:
   w b x a y f c e d j g i h n k m l r o q p v s u t z
6. La première lettre 's' est à la 19ème position de l'alphabet
   La lettre à la 19ème position dans l'alphabet changé est la lettre 'o'
7. Changements $(2;4), (5;5)$ et $(4;2)$, l'alphabet normal devient:
   u b y w d f c a h j g e l n k i p r o m t v s q x z
8. La deuxième lettre 'a' est à la 1ère position de l'alphabet
   La lettre à la 1ère position dans l'alphabet changé est la lettre 'u'
9. Changements $(4;2), (3;3)$ et $(2;4)$, l'alphabet normal devient:
   w b x a y f c e d j g i h n k m l r o q p v s u t z
10. La troisième lettre 'l' est à la 12ème position de l'alphabet
    La lettre à la 12ème position dans l'alphabet changé est la lettre 'i'
11. Changements $(2;4), (5;5)$ et $(4;2)$, l'alphabet normal devient:
    u b y w d f c a h j g e l n k i p r o m t v s q x z
12. La quatrième lettre 'u' est à la 21ème position de l'alphabet
    La lettre à la 21ème position dans l'alphabet changé est la lettre 't'
13. Changements $(4;2), (3;3)$ et $(2;4)$, l'alphabet normal devient:
    u b y w d f c a h j g e l n k i p r o m t v s q x z
14. La cinquième lettre 't' est à la 20ème position de l'alphabet
    La lettre à la 20ème position dans l'alphabet changé est la lettre 'q'

15. Le message crypté est ces cinq lettres trouvées: o, u, i, t et q ou ouitq

Décryptage

16. Les étapes 1-5 sont refaites avec les mêmes nombres pour la clé (43532)

17. Lettres du message crypté trouvées dans les alphabets avec changements

18. La première lettre 'o' est à la 19ème position de l'alphabet changé
    La lettre à la 19ème position dans l'alphabet normal est la lettre 's'

19. La deuxième lettre 'u' est à la 1ère position de l'alphabet changé
    La lettre à la 1ère position dans l'alphabet normal est la lettre 'a'

20. La troisième lettre 'i' est à la 12ème position de l'alphabet changé
    La lettre à la 12ème position dans l'alphabet normal est la lettre 'l'

21. La quatrième lettre 't' est à la 21ème position de l'alphabet changé
    La lettre à la 21ème position dans l'alphabet normal est la lettre 'u'

22. La cinquième lettre 'q' est à la 20ème position de l'alphabet changé
    La lettre à la 20ème position dans l'alphabet normal est la lettre 't'

23. Le message décrypté est ces cinq lettres trouvées: s, a, l, u et t ou salut

2.3 Codage

Cette partie contient l'explication de la procédure du codage de l'algorithme TCP dans un système informatique. Tout d'abord il faut savoir que dans l'informatique il y a énormément de langages de programmation, comme les différentes langues dans le monde (avec leur grammaire et orthographe propres). L'algorithme TCP a été codé à l'aide du langage Java car c'est un langage relativement simple et à usages multiples. L'apprentissage de ce langage a été fait à l'aide du site *openclassrooms.com* [5], qui est un site d'apprentissage qui propose des cours divers sur des sujets variés, la majorité sont en rapport avec l'informatique.
Le cours spécifique pour l'apprentissage du Java contient les connaissances nécessaires pour débuter la programmation et pour coder l'algorithme TCP. Ce cours prend environ 40 heures à compléter en entier avec de la théorie et des exercices pour tester les connaissances apprises. Le fait que l'algorithme TCP soit nouveau a rendu ce travail compliqué pour deux raisons différentes. En premier, même si le cours sur le Java contenait suffisamment d'informations pour programmer des systèmes de base, il manquait certains points

pour parvenir à coder l'algorithme dans son entièreté. La deuxième raison concerne l'originalité de ce projet, le fait qu'il n'y ait pas d'autres travaux sur le même thème implique qu'il n'est pas possible de faire de recherches précises pour aider à progresser.
Le système sur lequel est codé l'algorithme TCP fonctionne; il crypte et ensuite décrypte des messages selon les étapes du point précédent. La seule contrainte est que ce système n'accepte ni les marques de ponctuation (hormis les majuscules et les espaces), ni les spécialités linguistiques telles que les accents et les cédilles.

2.4 Transmission d'un message

Le point faible de l'algorithme TCP est sa clé de cryptage/décryptage. Le fait qu'il n'y ait pas de clé publique implique que la clé secrète doit être transmise entre celui qui écrit le message et celui qui le lit. Si une interception de la clé devait arriver durant la transmission, la sécurité des messages serait compromise car l'intercepteur pourrait alors décrypter tous les messages écrit à l'aide de la même clé. Ce problème pourrait être résolu en définissant un système avec une clé publique et on ne la transmettrait pas. Avec ce genre de système, il suffit de connaitre la clé publique de quelqu'un pour lui envoyer un message car seul lui pourra le décrypter grâce à sa clé privée qu'il garde secrète.

3 L'algorithme RSA

3.1 Fonctionnement

L'algorithme RSA a été inventé en 1977; il est connu car il est extrêmement sécurisé. De plus, cet algorithme n'a que très peu d'étapes et celles-ci ne sont pas très compliquées. Ce qui rend ce système sécurisé est l'utilisation de nombres très grands qu'il faudrait factoriser en nombres premiers de façon rapide pour le décrypter. En ce moment, nous ne possédons pas de moyens suffisamment rapides pour factoriser ces nombres gigantesques.
Voici un résumé des étapes dont consiste l'algorithme RSA:

1. Choisir deux nombres premiers p et q.
2. Calculer $n = p * q$
3. Calculer $\varphi(n) = (p-1) * (q-1)$

4. Choisir e avec $pgdc(e, \varphi(n)) = 1$
5. Calculer d avec $e * d \equiv 1 \mod \varphi(n)$ et $1 < e < \varphi(n)$
6. Définir la clé publique (n, e) et la clé secrète (p, q, d)
7. Numériser le message à l'aide du tableau suivant

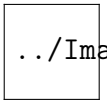

Figure 8: Tableau de chiffrage
La ligne du haut de chaque case représente de quelle lettre de l'alphabet il s'agit, le nombre se trouvant dessous représente son équivalent numérique par lequel il faut le remplacer pour chiffrer cette lettre

8. Crypter le texte $c \equiv m^e \mod n$ avec $m =$ message chiffré
9. Pour décrypter $m = c^d \mod n$

Exemple

Comme pour l'algorithme TCP, voici le déroulement entier du cryptage et décryptage d'un message à l'aide de l'algorithme RSA. Cet exemple est tiré du cours RSA donné en maths par Mme Reymond [4].

Cryptage

1. $p = 11$ et $q = 23$
2. $n = p * q = 253$
3. $\varphi(n) = (p - 1) * (q - 1) = 220$
4. $e * d \equiv 1 \mod \varphi(n)$ et $1 < e < \varphi(n) \Rightarrow e = 3$ (e le plus petit possible)
5. $1 < d < \varphi(n) = 220$ et $3 * d \equiv 1 \mod 220 \Rightarrow d = 147$
6. Clé publique (253,3) et clé secrète (11,23,147)
7. Message à crypter: salut

- Message numérisé: 18 00 11 20 19

8. S: $c \equiv m^e \bmod n \Rightarrow c \equiv 18^3 \bmod 253 \Rightarrow c = 13$
   A: $c \equiv 0^3 \bmod 253 \Rightarrow c = 0$
   L: $c \equiv 11^3 \bmod 253 \Rightarrow c = 66$
   U: $c \equiv 20^3 \bmod 253 \Rightarrow c = 157$
   T: $c \equiv 19^3 \bmod 253 \Rightarrow c = 28$

   - Message crypté: 13 0 66 157 28

Décryptage

1. $m \equiv c^d \bmod n$:
   13: $m \equiv 13^{147} \bmod 253 \Rightarrow m = 18$
   13: $m \equiv 0^{147} \bmod 253 \Rightarrow m = 0$
   13: $m \equiv 66^{147} \bmod 253 \Rightarrow m = 11$
   13: $m \equiv 157^{147} \bmod 253 \Rightarrow m = 20$
   13: $m \equiv 28^{147} \bmod 253 \Rightarrow m = 19$

2. Message décrypté: 18 0 11 20 19

3. Message déchiffré: s a l u t ou salut

3.2 Codage

Malgré le fait que le fonctionnement de l'algorithme RSA n'est pas trop complexe, son implémentation dans un système informatique est plus compliquée. Cela est dû au manque de fonctions mathématiques dans le langage du Java, ce qui oblige à tout d'abord coder les implémentations de ces fonctions qui seront nécessaires plus tard.
Les essais étant basés uniquement sur les connaissances de l'algorithme RSA acquises en cours de maths durant la deuxième année de gymnase en mathématiques et physique [4] et ceux de la programmation obtenues de façon individuelle à l'aide du site *openclassrooms.com* [5], ils n'ont pas fonctionnés. La cause de ces problèmes était que les choses apprises sur la programmation ne suffisaient pas à créer des implémentations personnelles pour les fonctions qu'emploie l'algorithme RSA. Le programme fini peut crypter des messages avec des nombres p et q allant de 0 à 1000.
Évidemment, ce programme n'est pas aussi sécurisé que celui qu'utilisent les vrais programmeurs, la raison de ceci est que les nombres utilisables dans ce programme sont beaucoup plus petits que ceux d'autres programmes et donc plus facilement factorisables.

4 Ce travail en général

4.1 Les principales difficultés rencontrées

Comme il est brièvement expliqué avant, ce qui a rendu ce travail difficile est le fait que l'algorithme TCP était un nouvel algorithme et en conséquent il n'y avait pas d'autres documents sur lesquels s'appuyer.
Ce projet consistait à partir sur une base semblant correcte selon les connaissances acquises durant ce travail et ensuite de regarder les erreurs qu'affichait le programme après la compilation du code. La grande majorité de ces erreurs ne venaient non pas de comment le programme était construit mais de l'utilisation maladroite de la syntaxe de Java. Ceci met en évidence à quel point ce langage a des utilisations variées mais cela donne aussi un aperçu de la quantité qu'il y aurait encore à apprendre sur ce sujet.

4.2 Techniques employées pour surmonter ces difficultés

La méthode principale employée pour résoudre les problèmes rencontrés était de regarder les erreurs que signalait le programme informatique, et ensuite, si les erreurs étaient connues, elles étaient corrigées directement sans aide externe. Par contre si ces erreurs étaient inconnues, l'aide d'Internet était utilisée pour en comprendre le sens. Lorsque les problèmes étaient identifiés et compris, soit ils étaient résolus directement, soit des informations supplémentaires étaient recherchées pour résoudre les problèmes spécifiques.

5 Conclusion

5.1 Algorithme TCP

En ce qui concerne l'algorithme TCP, l'aboutissement de son codage a été un moment d'énorme satisfaction pour moi car cela a été un travail très long et compliqué. En regardant toutes les étapes traversées pour en arriver là, j'ai remarqué à quel point mes idées et mes méthodes ont changées depuis le début de ce projet. Par exemple, la première version de mon code pour l'algorithme TCP comprenait plus de 8000 lignes mais maintenant qu'il est raffiné et terminé il en fait 650. Ce changement illustre très bien la différence de connaissances entre le début et la fin de ce travail. C'est une bonne chose car cela démontre que ce travail était une expérience pour apprendre de nouvelles choses et peut-être même de nouvelles manières de réfléchir.

5.2 Mon TM

En conclusion, ce projet m'a plu dans tous ses aspects et mon but a été atteint. J'ai pu inventer un nouvel algorithme de cryptage qui est l'algorithme TCP. Ensuite je suis parvenu à coder un programme informatique pour l'implémenter. Et la dernière partie du but, qui était de coder une implémentation de l'algorithme RSA, a été réussi et cela m'a donné une vue intérieure de la sécurité informatique d'aujourd'hui.
J'ai aimé ce projet pour les raisons suivantes; premièrement, il m'a permis d'apprendre de nouvelles choses sur toutes sortes de thèmes telles que la cryptographie et la programmation, d'où le titre de ce travail. Ensuite, ce TM m'a montré précisément ce que c'est de réaliser un gros projet du début à la fin. Cela est une très bonne expérience qui ne se présente que très rarement, en tout cas au gymnase. Finalement je suis content d'avoir réalisé ce projet sur le thème de la cryptographie et de la programmation car pour une fois, dans le cadre du gymnase, j'ai pu m'attarder sur quelque chose que j'ai beaucoup aimé.

6 Bibliographie

Cryptographie:
[1] Martignoni Nicolas, Cryptologie. Genève: Cahiers de la CRM, 2004, 46p.
[2] *en.wikipedia.org*, écrit en collaboration. "Enigma machine" (25 septembre 2018)
[3] *plus.maths.org*, Claire Ellis. "Exploring the Enigma" (1 mars 2005)
[4] Cours sur l'algorithme RSA suivi en maths avec Mme Reymond

Programmation:
[5] OpenClassrooms, Apprenez à programmer en Java,
https : *//openclassrooms.com/fr/courses/*26832*−apprenez−a−programmer−en − java*(site mis à jour en 2018, consulté le 4 octobre 2018)
[6] *www.dreamincode.net*, utilisateur chriscastro97. "For Loop With Multiple Conditions" (24 mars 2017)
[7] *codedost.com*, auteur inconnu. "Java Program on RSA Algorithm" (18 août 2018)
[8] *en.wikipedia.org*, écrit en collaboration. "Modular arithmetic" (24 septembre 2018)

Latex:
[9] Overleaf, Documentation,
https : *//www.overleaf.com/learn/latex/Main*$_P$*age* (site mis à jour en 2018, consulté le 27 septembre 2018)
[10] stack overflow, Top Questions, *https* : *//stackoverflow.com* (site mis à jour en 2018, consulté le 27 septembre 2018)
[11] *forums.techguy.org*, utilisateur theFAst0ne. "Solved: Using charAt() with an int in Java" (9 février 2011)
[12] *latex.org*, utilisateur LY4. "Problem with Inserting Pictures." (3 juin 2014)
[13] StackExchange, Explore Our Questions, *tex.stackexchange.com* (site mis à jour en 2018, consulté le 3 octobre 2018)