UNIVERSITÉ DU BURUNDI
GRENIER DU SAVOIR DU BURUNDI
Dépôt institutionnel officiel
https://repository.ub.edu.bi

Etude de la compilation de l'implémentation de l' algorithme du cryptosystème de Diffie Hellman

Hakizimana, Jean Bosco

2018-11
UB, FSI
https://repository.ub.edu.bi/handle/123456789/128

Document téléchargé depuis le **Grenier du Savoir du Burundi** — https://repository.ub.edu.bi

UNIVERSITE DU BURUNDI



FACULTE DES SCIENCES DE L'INGENIEUR

Département des Technologies de l'Information et de la Communication
Mémoire de Mastère en Génie Informatique

ETUDE DE LA COMPILATION DE L'IMPLEMENTATION DE L'ALGORITHME DU CRYPTOSYSTEME DE DIFFIE HELLMAN

Par :

HAKIZIMANA JEAN BOSCO

En vue de l'obtention d'un diplôme de Mastère en Génie Informatique

Défendu et soutenu le 22/11/2018

Sous la direction de :

Dr NDAYISABA Longin

Membres du Jury:

Dr NDIKUMAGENGE Jérémie : Président

Dr SAHINGUVU William : Vice-président

Dr MUKESHIMANA Michèle : Secrétaire

Dr NDAYISABA Longin : Directeur

ANNEE ACADEMIQUE : 2015-2016

DEDICACES

A ma chère Mère;

A mon cher Père;

A mes frères et sœurs;

A la famille de BIRURA Joseph ;

A la famille de CIZA François ;

A toute ma famille ;

A tous mes camarades de classe;

A tous mes amis.

REMERCIEMENTS

Mes vifs remerciements s'adressent premièrement à Dieu tout puissant qui ne m'a pas abandonné tout au long de mes études. Je remercie également mes parents et toute ma famille qui m'ont guidé depuis mon enfance jusqu'aujourd'hui. Mes remerciements s'adressent également à tous les enseignants qui m'ont enseigné depuis l'école primaire jusqu'à l'université, et plus particulièrement à mon Directeur de Mémoire Dr NDAYISABA Longin pour avoir dirigé ce mémoire. Je tiens à remercier tous les membres du Jury .Je remercie mes camarades de classe et tous mes amis qui m'ont aidé dans pas mal de choses tout au long de mon parcours académique.

LISTE DES SIGLES ET ABREVIATIONS

Bit : Binary digit

ECDSA : Elliptic Curve Digital Signature Algorithm

GF : Galois Field

ICT : Information and Communications Technology

LALR : Look ahead Left to right,Rightmost

LL : Left to right,Leftmost

LR : Left to right,Rightmost

Mod : Modulo

RSA : Rivest Shamir Adleman

SLR : Simple Left to right,Rightmost

LISTE DES TABLEAUX

Tableau 1 Echange de clé sans espion....17

Tableau 2 Echange de la clé avec présence de l'espion....18

Tableau 3 Pas de géant pour les courbes elliptiques....31

Tableau 4 Pas de bébé les courbes elliptiques....31

Tableau 5 pas de bébé pour le cryptosystème Diffie-Hellman....36

Tableau 6 Pas de géant pour le cryptosystème Diffie-Hellman....36

Tableau 7 Lexèmes et unités lexicales correspondantes....48

Tableau 8 Table de transition....49

Tableau 9 Table de prédiction....57

Tableau 10 Table de réduction....58

Tableau 11 Table d'action pour la méthode LR....67

Tableau 12 Table de successeur pour la méthode LR....67

Tableau 13 Table d'action pour la méthode SLR....73

Tableau 14 Table de successeur pour la méthode SLR....74

Tableau 15 Attribut valeur....83

Tableau 16 Attribut type....84

Tableau 17 Production et actions sémantiques correspondantes....86

LISTE DES FIGURES ET SCHEMAS

Figure 1 Addition de 2 points distincts 22

Figure 2 Doublement D'un point 23

Figure 3 Automate fini déterministe 49

Figure 4 Automate minimisé 51

Figure 5 Automate minimisé et renommé 52

Figure 6 Arbre syntaxique 59

Figure 7 Premier état de la pile 61

Figure 8 Deuxième état de la pile 61

Figure 9 Réduction de id en T 61

Figure 10 Troisième état de la pile 61

Figure 11 Réduction de T en E 62

Figure 12 Quatrième état de la pile 62

Figure 13 Précédent état de l'arbre syntaxique 62

Figure 14 Cinquième état de la pile 62

Figure 15 Etat précédent de l'arbre syntaxique 63

Figure 16 Sixième état de la pile 63

Figure 17 Réduction de id en T 63

Figure 18 Etat 7 de la pile 64

Figure 19 Réduction de T + E en E 64

Figure 20 Arbre syntaxique complet 64

Figure 21 Automate 66

Figure 22 Premier état de la pile 68

Figure 23 Deuxième état de la pile 68

Figure 24 Etat 3 de la pile 68

Figure 25 Etat 4 de la pile 69

Figure 26 Etat 5 de la pile 69

Figure 27 Réduction de T en E 69

Figure 28 Etat 6 de la pile 69

Figure 29 Etat 7 de la pile 69
Figure 30 Etat 8 de la pile 70
Figure 31 Etat 9 de la pile 70
Figure 32 Réduction de id en T 70
Figure 33 Etat 10 de la pile 70
Figure 34 Etat 11 de la pile 70
Figure 35 Réduction de E+T en E 71
Figure 36 Etat12 de la pile 71
Figure 37 Arbre syntaxique obtenue par LR 71
Figure 38 Automate SLR 72
Figure 39 Etat 1 de la pile 74
Figure 40 Etat 2 de la pile 75
Figure 41 Etat 3 de la pile 75
Figure 42 Réduction de id en F 75
Figure 43 Etat 4 de la pile 75
Figure 44 Etat 5 de la pile 75
Figure 45 Réduction de F en T 76
Figure 46 Etat 6 de la pile 76
Figure 47 Etat 7 de la pile 76
Figure 48 Réduction de T en E 76
Figure 49 Etat 8 de la pile 77
Figure 50 Etat 9 de la pile 77
Figure 51 Etat 10 de la pile 77
Figure 52 Etat 11 de la pile 77
Figure 53 Réduction de id en F 77
Figure 54 Etat 12 de la pile 78
Figure 55 Etat 13 de la pile 78
Figure 56 Réduction de F en T 78

Figure 57 Etat 14 de la pile 78
Figure 58 Etat 15 de la pile 78
Figure 59 Etat 16 de la pile 78
Figure 60 Etat 17 de la pile 79
Figure 61 Réduction de F en T 79
Figure 62 Etat 18 de la pile 79
Figure 63 Etat 19 de la pile 79
Figure 64 Réduction de T*F en T 80
Figure 65 Etat 20 de la pile 80
Figure 66 Etat 21 de la pile 80
Figure 67 Réduction de E+T en E 80
Figure 68 Etat 22 de la pile 81
Figure 69 Arbre syntaxique obtenue par SRL 81
Figure 70 Arbre syntaxique du mot 82
Figure 71 Arbre syntaxique abstrait du mot analysé 82
Figure 72 Attributs synthétisés 84
Figure 73 Attribut hérité 85
Figure 74 Arbre syntaxique décoré 86
Figure 75 Arbre syntaxique abstrait pour la génération du code 87
Figure 76 Manipulation des variables par des registres 88

RESUME DU THEME

Le thème de notre projet est intitulé l'étude de la compilation de l'implémentation de l'algorithme du cryptosystème de Diffie-hellman. Notre projet concerne d'une part le fonctionnement d'un compilateur pour générer un code cible en langage de bas niveau à partir d'un programme de haut niveau pour les développeurs des logiciels. Pour ce faire toutes les phases de la compilation sont détaillées. La phase d'analyse lexicale qui permet de déterminer les mots qui sont reconnus dans un langage donné, les automates sont utilisés pour reconnaître ces mots. Pour la phase d'analyse syntaxique, les deux méthodes d'analyse syntaxique qui sont l'analyse syntaxique ascendante et l'analyse syntaxique descendante permettent de ranger les mots tout en respectant les règles de la grammaire. Dans cette phase les grammaires hors contextes sont mises en jeux pour déterminer si la succession des mots est reconnu par une grammaire du langage. La phase d'analyse sémantique qui est destinée à donner le sens, les grammaires attribuées sont employées. La phase de synthèse donnant lieu à un code cible. D'autre part notre projet concerne à explorer les systèmes cryptographiques à échange sécurisé des clés pour lesquels la sécurité est basée sur la difficulté de résoudre le problème du logarithme discret et la mise en jeu des grands nombres premiers notamment pour le cryptosystème El Gamal, la signature numérique El Gamal, la cryptographie et la signature numérique basées sur les courbes elliptiques et le cryptosystème de Diffie hellman. La cryptographie quantique est aussi explorée. Pour ce dernier la sécurité découle du théorème du non-clonage et du principe d'incertitude de Heisenberg. La signature numérique RSA et la fonction de hachage sont utilisées pour sécuriser les échanges des paramètres de l'algorithme de Diffie Hellman.

TABLE DES MATIERES

DEDICACES i

REMERCIEMENTS ii

LISTE DES SIGLES ET ABREVIATIONS iii

LISTE DES TABLEAUX iv

LISTE DES FIGURES ET SCHEMAS v

RESUME DU THEME viii

TABLE DES MATIERES ix

CHAPITRE I INTRODUCTION GENERALE 1

I.1 Introduction 1

I.2Actualité et justification du projet 1

I.3. Objectifs du projet 2

I.3.1. Objectif global 2

I.3.2. Objectifs spécifiques 2

I.4. Problématiques et solutions proposées 2

I.4.1. Problématiques 2

I.4.2. Solutions proposées 3

I.5. Résultats attendus 3

I.6. Impacts socio-économiques 3

I.7. Domaine d'application 3

I.8.Méthodologie 4

I.9.Conclusion 4

CHAPITRE II ETUDE EXPLORATOIRE DES SYSTEMES CRYPTOGRAPHIQUES D'ECHANGE SECURISE DES CLES 5

II. 1. Introduction 5

II.2. Cryptosystème de Diffie-Hellman 5

II.3.Cryptosystème D'ElGamal 7

II.4 Protocole de signature numérique El Gamal 13
II.5. Cryptographie quantique 14
II.5. 1. Généralités 14
II.5. 2. Mécanisme de mesure de la polarisation d'un photon 15
II.5. 3. Echange des clés 15
II.6 .Cryptographie basée sur les courbes elliptiques. 19
II.6 .1.Généralités 19
II.6 .2. Courbes elliptiques dans un corps binaire 19
II.6. 3. Addition de deux points dans $GF(2^n)$ 21
II.6 .4. Addition de deux points de la courbe dans Zp 21
II.6.5.Théorème de Hasse 23
II.6 .6. Echange de clé basée sur les courbes elliptiques 24
II.6 .7. Algorithme de cryptage et de décryptage 25
II.6.8 Algorithme de signature numérique 28
II.6 .9.Algorithmes résolvant le problème du logarithme discret 30
II.10. Conclusion 31
CHAPITRE III IMPLEMENTATION DU CRYPTOSYSTEME DE DIFFIE HELLMAN 32
III.1. Introduction 32
III.2. But et fonctionnement de l'algorithme Diffie-Hellman 32
III.3.Problème du logarithme discret 33
III.3.1. Problème du logarithme discret dans le group Z_p* 33
III.3.2.Quelques propriétés de Z_P* 33
III.3.3.Problème du logarithme discret dans le groupe Z_n* 34
III.3.4. Quelques algorithmes utilisés pour résoudre le problème du logarithme discret. 35
III.4. Diffie-Hellman et l'attaque de l'homme du milieu 36
III.5. Signature numérique 37
III.6. Modèle mathématique 44
III. 7.Conclusion 46

CHAPITRE IV ETUDE DE LA COMPILATION 47
IV.1. Introduction 47
IV.2. Analyse lexicale 47
IV.2.1. Automate fini déterministe 48
IV.2.2.Minimisation de l'automate 49
IV.2.3. Expression régulière 51
IV.3.Analyse syntaxique 54
IV.3.1. Méthode universelle 54
IV.3.2.Analyse syntaxique descendante. 55
IV.3.3.Analyse syntaxique ascendante 59
IV.4.Analyse sémantique 81
IV.4.1 Arbre syntaxique décoré et l'arbre syntaxique abstrait 81
IV.4.2 Traduction dirigée par la syntaxe 83
IV.5. Phase de synthèse 85
IV.5.1 Génération du code à partir du code en trois adresses. 85
IV.5.2.Génération du code à partir de l'arbre syntaxique abstrait. 87
IV.6.Conclusion 88
CONCLUSION GENERALE ET RECOMMANDATIONS 89
CONCLUSION GENERALE 89
Recommandations 90
Recommandations adressées aux entreprises 90
Recommandations aux futurs lauréats 90
REFERENCES 91
OUVRAGES 91
WEBOGRAPHIE 92

CHAPITRE I INTRODUCTION GENERALE

I.1 Introduction

Le domaine informatique a beaucoup révolutionné le monde permettant aux entreprises l'utilisation des nouvelles technologies de l'information et de la communication au sein des entreprises. Les technologies de l'information et de la communication ont un rôle important dans plusieurs domaines de la société en général et dans le domaine économique en particulier. Le transfert, le traitement de l'information sont rendus rapide et facile grâce aux nouvelles technologies de l'information et de la communication. C'est dans cette optique que l'être humain ne cesse pas de rendre les nouvelles technologies de l'information et de la communication plus meilleures avec le temps. Les programmes exécutant des tâches spécifiques sont développés pour bien traiter et transmettre l'information.Comme des informations confidentielles circulent sur les réseaux, la sécurité des informations est devenue une préoccupation importante des utilisateurs. La cryptographie étant l'un des moyens de sécurité de l'information pendant la transmission de ces informations, dans notre travail nous utilisons le cryptosystème de Diffie-hellman pour que l échange de l'information via un réseau public soit fait avec garantie de l'intégrité, de confidentialité de l'information et de l'authenticité et la non-répudiation.

Notre projet se rapporte surtout sur l'étude de la compilation de l'algorithme de ce cryptosystème de Diffie-hellman permettant de voir en détail le fonctionnement du compilateur utilisé pour compiler l'algorithme de ce cryptosystème en particulier et le fonctionnement du compilateur de façon générale pour compiler n'importe quel programme en langage de haut niveau.

I.2Actualité et justification du projet

A l'heure actuelle avec l'avancée spectaculaire dans le domaine des nouvelles technologies de l'information et de la communication, l'information circulant sur le réseau peut être interceptée ou modifiée par les malfaiteurs. Dans des entreprises il est commode de ne pas exposer une information aux personnes qui ne sont pas destinée à avoir connaissance de cette information. Pour une entreprise qui utilise un réseau public pour échanger des informations, il est crucial d'implanter la cryptographie symétrique utilisant la clé obtenue du cryptosystème de Diffie Hellman. Les programmeurs ont un intérêt majeur de comprendre les détails sur le fonctionnement du compilateur.

En matière de sécurité informatique, quelques entreprises utilisent des antivirus et l'accès aux outils informatiques avec des mots de passe, mais n'ont pas intégré la cryptographie dans leur système d'échange d'information.

Du point de vue de services de développement de logiciels, il y en a des développeurs disposant des outils bien adaptés à la programmation moderne que ce soit au point de vue matérielle qu'au point de vue logicielle.

I.3. Objectifs du projet

Notre projet spécule surtout sur l'objectif particulièrement bien spécifié, bien que ces objectifs puissent se globaliser.

I.3.1. Objectif global

L'objectif global de notre travail est de mettre en place un cryptosystème permettant l'échange des clés et de montrer en détail le fonctionnement d'un compilateur au moment de la compilation du programme source.

I.3.2. Objectifs spécifiques

Objectifs spécifiques qu'on doit atteindre dans notre projet sont :

- Générer des clés par le cryptosystème de Diffie Hellman en utilisant des grands nombres premiers s'écrivant au moins sur 2048 bits.
- Echanger des paramètres permettant la génération de ces clés de façon sécurisée.
- Rendre confidentielles et intègres les clés échangées
- Définir et montrer en long et en large les étapes et le fonctionnement du compilateur.

I.4. Problématiques et solutions proposées

I.4.1. Problématiques

D'une part en échangeant des informations, on prétend que l'information arrive à la destination sans être altérée ou sans qu'une personne non autorisée puisse récupérer et regarder le contenu de l'information. Par ailleurs, en utilisant le réseau internet accessible par tout le monde pour se communiquer une clé, les informations échangées peuvent être interceptées et altérées par des personnes malintentionnées .D'autre part, le fonctionnement du compilateur n'est pas été bien maîtrisé par pas mal de développeurs des logiciels.

I.4.2. Solutions proposées

Pour bien mener le service de production des logiciels, les développeurs ont intérêt majeur de bien maîtriser le fonctionnement d'un compilateur. Pour que l'information soit sécurisée contre les attaques pouvant intercepter ou altérer les informations au moment de la communication, des mesures de sécurisé suivantes doivent être prises :

1. Générer une clé de Diffie Hellman qui est utilisée pour le chiffrement et le déchiffrement des données dans un cryptosystème symétrique.

2. La signature des paramètres échangés pour générer la clé de Diffie Hellman est réalisée en utilisant une fonction de hachage et le schéma du cryptosystème RSA avec des nombres premiers assez grands s'écrivant au moins sur 2048 bits pour pallier aux attaques dites Man in the middle.

I.5. Résultats attendus

Les clés éphémères de Diffie Hellman seront générées chaque fois qu'il est question d'échanger des données cryptées. Les paramètres échangés entre deux individus pour générer la clé sont signés. Connaissant le fonctionnement du compilateur au moment de la compilation du programme écrit en langage de haut niveau, les développeurs produiront mieux beaucoup de logiciels qu'avant. Les développeurs seront également capables de développer leurs propres compilateurs qui seront utilisées pour des faits particuliers selon les demandes de leurs clients.

I.6. Impacts socio-économiques

La qualité des services rendus par les développeurs sera très considérable. Du point de vue service de production des logiciels, les développeurs seront menés à utiliser peu de temps pour répondre aux besoins de leurs clients. Ayant une communication sécurisée avec des clés obtenues à partir des grands nombres premiers, mais aussi la signature avec le cryptosystème RSA et la fonction de hachage, les attaquants sont incapables d'intercepter ou d'altérer les informations. Ce qui permettra de travailler avec la tranquillité, par conséquent de gagner la confiance de la clientèle.

I.7. Domaine d'application

Le produit du travail de notre projet est destiné à être utilisé dans tous les domaines où il s'avère nécessaire de crypter les informations communiquées entre deux individus nécessitant d'abord l'échange sécurisé des clés. Le projet est aussi de grande importance pour n'importe quel développeur

des logiciels qui veut bien mener son métier de développement des logiciels ou de produire des compilateurs.

I.8.Méthodologie

La méthode d'entretien est utilisée pour collecter et analyser les données. L'entretien avec le chef du personnel de l'une des entreprises des développeurs a été fait.

I.9.Conclusion

Avec les besoins et l'utilisation des technologies de l'information et de la communication, notre projet est de grande importance dans la société en général et dans les entreprises qui développent des logiciels en particulier qui en profite de deux côtés : côté de développement des logiciels et côté de sécurisation de la communication en utilisant des cryptosystèmes très puissants.

CHAPITRE II ETUDE EXPLORATOIRE DES SYSTEMES CRYPTOGRAPHIQUES D'ECHANGE SECURISE DES CLES.

II. 1. Introduction

La cryptographie est une science permettant de convertir des informations "en clair" en informations codées, c'est à dire non compréhensibles, puis, à partir de ces informations codées, de restituer les informations originales [2.1]. L'échange de clé est un problème majeur auquel la cryptographie symétrique est confrontée .Si deux personnes veulent communiquer des messages cryptés, ils doivent se rencontrer pour échanger la clé, ce moyen est inefficace si deux personnes sont séparées par une longue distance et qu'ils veulent changer la clé chaque fois qu'ils veulent échanger des messages. En cryptographie asymétrique, la clé privée peut être trouvée à partir de la clé publique lorsque cette dernière est de petite taille. Le cryptosystème de Diffie-Hellman, Cryptosystème D'El Gamal, la cryptographie quantique et la cryptographie basée sur les courbes elliptiques sont traités dans notre travail comme moyen efficace d'échange des clés.

II.2. Cryptosystème de Diffie-Hellman

Ce cryptosystème n'est pas utilisé pour crypter un message mais il est utilisé pour générer et échanger la clé de manière sécurisée qui sera utilisée ultérieurement dans un autre cryptosystème et plus précisément le cryptosystème symétriques. Si deux personnes Alice et Bob veulent échanger la clé en utilisant le cryptosystème de Diffie-Hellman. Les algorithmes utilisés pour échanger les clés sont les suivants :On donne q d'au moins 160 bits de longueur; p d'au moins 1024 bits. On a besoin des paramètres publics (p, q, g). Les parties Alice et Bob doivent sélectionner les clefs privées dans l'intervalle [2, q − 2]

L'algorithme de génération des paramètres publics

ENTREE: nombres p et q premiers avec des bits de longueur exigée.

SORTIE: paramètres (p, q, g).

1. Générer au hasard un nombre premier q avec la longueur des bits exigée.

2. Choisir un nombre j de longueur des bits égale à bitlen(p) − bitlen(q)

3. Calculer p = jq + 1. Si p pas premier, alors aller à 2.

4. Choisir au hasard un nombre h dans l'intervalle $1 < h < p - 1$.

5. Calculer $g = h^{\frac{p-1}{q}} \bmod p$. Si $g = 1$ alors aller à 4.

6. Retourner (p,q,g)

Algorithme de vérification des paramètres publics

ENTREE: paramètres (p,q,g).

SORTIE: Accepter ou rejeter les paramètres.

1. Tester $1 < g < p - 1$. Si non, retourner "rejeter paramètres" et stop

2. Tester la primalité de q . Si q pas premier alors retourner "rejeter paramètres" et stop

3. Tester la primalité de p. si p pas premier alors retourner "rejeter paramètres" et stop

4. Calculer $(p-1) \bmod q$. Si $\neq 0$ alors retourner "rejeter paramètres et stop

5. Calculer $g^q \bmod p$. Si $\neq 1$ alors retourner "rejeter paramètres" et stop

6. Retourner "Accepter paramètres"

Algorithme de génération de la clef à partager par la partie Alice

ENTREE: Paramètres (p,q,g)

SORTIE: La clef privée/publique (a,A)

1. Alice choisit un nombre a dans l'intervalle $[2, q - 2]$.

2. Calcule $A = g^a \bmod p$.

3. Retourne (a,A). garde a secrète.

Algorithme de génération de la clef à partager par la partie Bob

ENTREE: Paramètres (p,q,g)

SORTIE: La clef privée/publique (b,B)

1. Bob choisit un nombre b dans l'intervalle $[2, q - 2]$.

2. Calcule $B = g^b mod p$.

3. Retourne (b,B). garde b secrète.

Algorithme de calcul de la clef secrète partagée par la partie Alice

ENTREE: Paramètres (p,q,g),B,a

SORTIE: clef secrète partagée Z_A.

1. Tester $1 < B < p$ *et* $B^a mod\ p = 1$. *Si non "échec" et stop.*

2. Calcule $Z_A = B^a mod\ p$

3. Retourner Z_A.

Algorithme de calcul de la clef secrète partagée par la partie Bob

ENTREE: Paramètres (pq,g),A,b

SORTIE: clef secrète partagée Z_B.

1. Tester $1 < A < p$ *et* $A^a mod\ p = 1$. *Si non "échec" et stop.*

2. Calcule $Z_B = A^b mod\ p$

3. Retourner Z_B.

La clé qui est générée est alors $Z_A = B^a \text{mod } p = Z_B = A^b \text{mod } p$ .L'importance de ce cryptosystème réside dans la difficulté de résoudre un problème appelé problème du logarithme discret qui consiste à déterminer la valeur de la variable x dans les situations suivantes : $Log_\alpha \beta = x$ ou $\beta = \alpha^x$ .Ce problème est difficile à résoudre voire impossible à résoudre en temps raisonnable pour les grands nombres. Néanmoins, ce cryptosystème peut être heurté à l'attaque dite, the man in the middle. Mais cette attaque peut être évitée en utilisant la signature numérique.

II.3.Cryptosystème D'ElGamal

Les algorithmes de génération et d'échange des clés du Cryptosystème D'ElGamal sont établis comme ceux du cryptosystème de Diffie-Hellman. Le Cryptosystème D'ElGamal permet de crypter et de décrypter un message en utilisant la clé échangée. Le cryptage et le décryptage sont procédés comme suit :

Algorithme de cryptage d'ElGamal

ENTREE: Paramètres (p,q,g); clef publique B; message à crypter m dans l'intervalle $0 < m < p - 1$

SORTIE: Ciphertext (C_1, C_2).

1. *Choisir au hasard k dans l'intervalle* $1 < k < p - 1$
2. *Calculer* $C_1 = g^k \bmod p$
3. *Calculer* $C_2 = mB^k \bmod p$
4. *Retourner ciphertext* (C_1, C_2).

Algorithme de décryptage d'ElGamal

ENTREE: Paramètres (p,q,g); clef privée b; ciphertext (C_1, C_2)

SORTIE: Message décrypté m.

1. *Calcule* $m = C_1^{p-b-1} C_2 \bmod p$.
2. *Retourne m.*

Une particularité du chiffrement d'ElGamal est qu'il n'est pas déterministe. En effet, pour des paramètres identiques, tels que p, g et la paire de clés,deux chiffrements du même message clair M donneront deux messages chiffrés différents.[1] Il est à remarquer que l'importance de ce cryptosystème réside dans la difficulté de résoudre le problème du logarithme discret .Mais également, il peut être soumis à l'attaque dite Man in the middle. Cette attaque peut aussi être contournée en utilisant la signature numérique pour assurer l'intégrité des messages et l'authentification de l'expéditeur au moment de l'échange des paramètres pour générer la clé.

Le cryptosystème El Gamal est illustré par un exemple ci-dessous :

Q=27177511748142749764433481485080332090511968347863080031260016852453688601265936681204648882214515935565463838346597120952597187768284534262666514210649978511969706615077707131590089854409963099468777693167943722573366993900900395914660278597269441541358086171267962570124459908749744910015726519979895299325390141388532269169778025122154616103754646812220333519777621439955474557840144444283375575598573951447344144460945171176670543608078607496906045196987742450448868982410113929303526887247949414251

631211354225276518285138386161815840541008678963431040449893295
252677966930572210688177601832937502294746066866523 9

La valeur de j trouvée est: 930

La valeur trouvée de P est:
252750859257727572809231377811247088441761305635126644290718156
727819303991773211135203234604594998200758813696623353224859153
846245046168642798582159044800161318271520222676323787835646012
656825059632546461876619932313043278373682006340590954605806334
630201392792051902157477151372627663146256635813026283726128314
913350103278935633636037929764918215353649101733931879391585913
387913343331835392853066737748460300543486790091943036055555131
049721226220331986004789174481536414059542522800051405929552583
702655942950716200517869913048873170313807143599086761840076458
499050924543215594000516970463187713411384218586722 71

La valeur de h est 2

La valeur de g est:
907603093553334388914833067718445166095739869176876500888532628
977014561255129622525127145078220428826781447625850203277865347
439907779362665301868348629532338239038359045333216971685689878
989788964352894501609622844084904100268608494323083708897755744
656436434414009291848967782 4

La valeur de a est 5
La valeur de b est 7

La valeur g^a mod p que l'interlocuteur A envoie à l'interlocuteur B est :
161060044139444141559818123108773550260866326345565982421420347
056017201775646673664243244475102561820236236553721945736476203
768496093071697774904725148355132551616182108357891451232633236
463904173879882633374847387094051119794192719094577421456934941
877443323440702667536603190595070417878932615640318530254539301
112756382027636431829280776869401192593606920582853320875603167
155044990326519883948240346848383483052837575759856388922949336
824622334649051823760750720553818832612524098969025888821200636
178077690276484864781795138863875508988152054724092138359879868
723127531395408845797099572184346963179323234931270 72

La valeur g^b mod p que l'interlocuteur B envoie à l'interlocuteur A est :

174962041614393904292556422847575948812068287368896754866858431
590118907069640271414376979096656988469531165381805897618991234
141534763531546537766670071906308929361989073492647590725031195
339489216391760579607338540256846170612017148020940714111291787
053196428139892063424220852472039768657376648560346533678836785
434885242876249619370301902507335286296259149773259559432361075
733112615713430449734159604221261673928856154301104540578693019
996872638423562334625088701787860561156922591233077279249966868
119998540465997581670864574885204564755431837400386139750118415
322680116737806383142710780577596965162534139755390 90

ZA=129839600980286442787206960784851245095973723704583450909730
295742315537072777280778529488970042737615270802453019792033470
973160561174761757958909876828259891611431742671176377660937557
974001836034792051506163640933438508961106520152122886671127776
972976138300112668690389506373998091991373595858198093443259435
711118250517438991370223536904407969511802554332343548360619779
669092397101939741443207366064389377543108932247670050057725746
602252990015409010581740646309816997703314062740124048361522161
730882049168792357128586594083705978832339123722106275142065186
484325700249924783366888923925245937819256307692956072419

ZB=129839600980286442787206960784851245095973723704583450909730
295742315537072777280778529488970042737615270802453019792033470
973160561174761757958909876828259891611431742671176377660937557
974001836034792051506163640933438508961106520152122886671127776
972976138300112668690389506373998091991373595858198093443259435
711118250517438991370223536904407969511802554332343548360619779
669092397101939741443207366064389377543108932247670050057725746
602252990015409010581740646309816997703314062740124048361522161
730882049168792357128586594083705978832339123722106275142065186
484325700249924783366888923925245937819256307692956072419

Soit hello le mot à crypter

Soit la valeur de k =4

Le cryptage du mot hello est constitué par des couples suivants :

(18229249207894241071152470062568689310171501954882944639580318
47045574970137633025486788701437048152536669108077357104385967 5
08693434392587592050424994366322848498782843490062082388232673 4
45658906310936775400980321436918616125814684257503878322023767 2
98113276276929226302649718833817605321494557420067252295853331 4
85391008999178641986042674993988136604809703379814201658398163 5
86883965011775053196119814011556000577443074766575501541143908 1
17155584763647023783583243420634270323189491962005179034868061 1
95102569556871887681055294897551860343725787879253139755855636 3
28619129038102212041333494608661801146581541192449628 1 ,

12126955728879389630530717171005872199848497390003153000994180 0
48207435251716438003487403191621734729962856998514946718982301 4
10711903477940015881808208461709856144890399437204402173141248 7
66960428674208391106730360983559835851222595277689592299266304 7
94284699966293298920820513641812582564222328794810301534606720 1
41322691770332341219702892903212222031655157871334729134286517 3
90267067352471030619326620875338588880076182049157809012358185 4
83509909849558882802552822183529176973911088146103703952608095 7
32803558621173961280908582118043977605783803127317986551561087 0
00592159326246311916419387422981484259322265519569765)

(18229249207894241071152470062568689310171501954882944639580318
47045574970137633025486788701437048152536669108077357104385967 5
08693434392587592050424994366322848498782843490062082388232673 4
45658906310936775400980321436918616125814684257503878322023767 2
98113276276929226302649718833817605321494557420067252295853331 4
85391008999178641986042674993988136604809703379814201658398163 5
86883965011775053196119814011556000577443074766575501541143908 1
17155584763647023783583243420634270323189491962005179034868061 1
95102569556871887681055294897551860343725787879253139755855636 3
28619129038102212041333494608661801146581541192449628 1 ,
15908644128413031023647074774526087908997042845865709119179163 9
54867728627013218243096857895976987544880178270521820375956471 1
92504988747679176554349032988393939535533795401032644796229464 8
74803902860292081605480964848766509559699437825246667740920841 8
56674010234058763698755606107274402560914341241903703243554777 5
92560452349786125388185160229084571070176677095697036899313791 4
34388716214073983273098097717130576805880980097319633380217504 3
90567567109423187877557467664952488252560021072274629704746702 5
12573793497642188752424477366399289689592656345938116700344228 5
24117256381484077051531124802966817528758672971865143)

(18229249207894241071152470062568689310171501954882944639580318
47045574970137633025486788701437048152536669108077357104385967 5
08693434392587592050424994366322848498782843490062082388232673 4
45658906310936775400980321436918616125814684257503878322023767 2
98113276276929226302649718833817605321494557420067252295853331 4
85391008999178641986042674993988136604809703379814201658398163 5
86883965011775053196119814011556000577443074766575501541143908 1
17155584763647023783583243420634270323189491962005179034868061 1
95102569556871887681055294897551860343725787879253139755855636 3
28619129038102212041333494608661801146581541192449628 1 ,
15509733171425453533349952960020487535709146970023966319771473 3
96921020884379838055181571405967897583431722425726560283845380 1
63195957990575894937825743919469788899749878907977538270211827 7
11729965080515019170950220240385713519042872092299856863920466 5
32105666035674409288822962066951078006841533392453308713546920 7
03450681107025150116261134127546363825414768963316048093969682 4
44368646981394940176066876344564614997119344321340144390397597 9
75757074377084208902706267357682642404363928337877516480741706 1
49865110217573531336149718888532667171165570289127371747852780 2
52527060737369811536287636598440630347119862869400018)

(18229249207894241071152470062568689310171501954882944639580318
47045574970137633025486788701437048152536669108077357104385967 5
08693434392587592050424994366322848498782843490062082388232673 4
45658906310936775400980321436918616125814684257503878322023767 2
98113276276929226302649718833817605321494557420067252295853331 4
85391008999178641986042674993988136604809703379814201658398163 5
86883965011775053196119814011556000577443074766575501541143908 1
17155584763647023783583243420634270323189491962005179034868061 1
95102569556871887681055294897551860343725787879253139755855636 3
28619129038102212041333494608661801146581541192449628 1 ,
15509733171425453533349952960020487535709146970023966319771473 3
96921020884379838055181571405967897583431722425726560283845380 1
63195957990575894937825743919469788899749878907977538270211827 7
11729965080515019170950220240385713519042872092299856863920466 5
32105666035674409288822962066951078006841533392453308713546920 7
03450681107025150116261134127546363825414768963316048093969682 4
44368646981394940176066876344564614997119344321340144390397597 9
75757074377084208902706267357682642404363928337877516480741706 1
49865110217573531336149718888532667171165570289127371747852780 2
52527060737369811536287636598440630347119862869400018)

(18229249207894241071152470062568689310171501954882944639580318
47045574970137633025486788701437048152536669108077357104385967 5
08693434392587592050424994366322848498782843490062082388232673 4
45658906310936775400980321436918616125814684257503878322023767 2
98113276276929226302649718833817605321494557420067252295853331 4
85391008999178641986042674993988136604809703379814201658398163 5
86883965011775053196119814011556000577443074766575501541143908 1
17155584763647023783583243420634270323189491962005179034868061 1
95102569556871887681055294897551860343725787879253139755855636 3
28619129038102212041333494608661801146581541192449628 1 ,
11728044771891812140233595356500271826560601514161410201586489 4
90260727509083057815572116701612644768514401153719686626871210 3
81402872720836734265284919392785705509106482944149295647123611 6
03886490894431328672199616375179039810566029544742781422265929 4
69716355767908944510887869601489258010149520945359907004598863 2
52212920527571365947778866801674014786893249738953740328942408 4
00246998119791987522295399502772627071314546273178320022538279 0
68699417117219903827701621876259331125714995411706590728603099 3
70094875341105303864633823640177355087356717070507241599069638 7
29001963682132046401175899218455297077683455417104640)

Le décryptage de ces couples donne le mot hello.

II.4 Protocole de signature numérique El Gamal

Ce protocole n'est pas déterministe, pour un message donné on peut avoir beaucoup de signatures valides. Prenons A comme expéditeur et B comme le récepteur du message. A possède la paire de clés (X_A, Y_A) , X_A est la clé privée de A et X_A n'est pas diviseur de P-1. $Y_A = g^{X_A} \bmod p$. B possède la paire de clés (X_B, Y_B). , X_B est la clé privée de B et X_B n'est pas diviseur de P-1. $Y_B = g^{X_B} \bmod p$. Les paramètres publics sont p, g et Y_A.

P est un nombre premier dont le logarithme discret est difficile à résoudre.

G est un générateur

La signature de A sur le message m est représentée par le couple (r, s).

$r = g^x \bmod p$

$s = [(m - X_A * r)/x] \bmod (p-1)$.

X est choisi aléatoirement dans Z_P chaque fois qu'un message doit être signé par A. X doit être différent de p-1.

Pour (m,r,s) ,B peut vérifier que (r,s) est la signature de A sur le message m en vérifiant l'égalité $g^m = Y_A{}^r * r^s \bmod p$.

Exemple d'illustration

P=547
g=9
X_A=23
Y_A trouvé est 81
R=304

m=100

S=172

g^m= 81

x=125

$Y^r{}_A * r^s \bmod p$= 81

La signature est validée.

II.5. Cryptographie quantique

II.5. 1. Généralités

Cette technique utilise la polarisation des photons de la lumière. La polarisation d'un photon consiste à donner l'orientation ou la direction du photon. La polarisation des photons est mesurée par un angle variant entre 0˚et 180 ˚.En cryptographie quantique quatre types de polarisation qui sont utilisées sont :

-La polarisation horizontale correspondant à l'angle de 0˚

-La polarisation verticale correspondant à l'angle de 90˚

-La polarisation diagonale droite correspondant à l'angle de 45˚

-La polarisation diagonale gauche correspondant à l'angle de 135˚

La lumière n'est pas polarisée à l'état naturel, elle est composée par plusieurs photons qui ont des directions variées.

En effet, la lumière étant composée de plusieurs photons, chacun de ces photons ayant sa propre direction, le polariseur horizontal arrête tous les

photons qui n'ont pas une polarisation horizontale et laisse passer ceux qui ont la polarisation horizontale. A la sortie du filtre, les photons obtenus sont seulement ceux polarisés horizontalement. Le même principe est appliqué pour obtenir des photons polarisés dans chacune des autres directions. Par convention un 1 est représenté par photon polarisé à 90˚ ou 135˚ et 0 est représenté par un photon polarisé à 0˚ ou 45˚.

II.5. 2. Mécanisme de mesure de la polarisation d'un photon

Ayant deux bases, celles permettant les polarisations horizontales ou verticales et la base permettant les polarisations diagonales gauches ou les polarisations diagonales droites, la première étape est le choix de la base à utiliser. Deuxièmement on utilise un filtre pour aiguiller l'une des directions parmi les 2 directions de la base. Pour la base utilisant les angles 0˚et 90˚, le filtre horizontal peut être utilisé. Dans ce cas, l'une de deux possibilités est observée à la sortie du filtre.

-Le photon est passé à travers le filtre, dans ce cas le photon est polarisé horizontalement.

- Le photon n'est pas passé à travers le filtre, dans ce cas le photon n'est pas polarisé horizontalement.

II.5. 3. Echange des clés

Si deux personnes Alice et Bob veulent générer une clé. Les photons polarisés sont mesurés à l'aide de l'une de deux bases pour chacun des bits qui sont envoyés par Alice à Bob via la fibre optique. Bob ne connaît pas la base et l'orientation utilisée par Alice pour chaque photon. Les photons émis par Alice sont reçus par Bob et sont mesurés en utilisant aléatoirement une de deux bases. Bob informe Alice les Bases utilisées pour chaque photon via un canal public et ils comparent le résultat. Seuls les photons pour lesquels les mêmes bases ont été appliquées sont retenus comme une clé. Pour le cas où ils n'ont pas utilisées la même base, les photons ne sont pas considérés. La sécurité de ce type d'échange des clés découle du théorème du non-clonage et du principe d'incertitude de Heisenberg qui nous montre qu'un espion ne peut ni copier ni voir les photons envoyés par Alice à Bob sans que Alice et Bob le constatent.

théorème : pour un état quantique inconnu quelconque, il n'est pas possible de faire une copie parfaite de cet état (sans détruire l'état initial) : théorème de non-clonage :[2 .2]

le principe d'incertitude montre qu'il est impossible d'attribuer à un corpuscule, à un instant donné, par exemple une position et une quantité de mouvement parfaitement déterminées. il est impossible de mesurer simultanément, avec une précision absolue, la position p et la vitesse q d'une particule dans le domaine de l'atome, car la mesure perturbe le système : le principe d'incertitude de heisenberg . [2.3]

. En effet, il faut utiliser des ondes très courtes afin de saisir la position de la particule. Or ces ondes sont, par exemple, des photons qui communiquent une énergie à la particule et change du même coup sa position.
Les spécificités de l'information quantique précise qu'une information quantique : on ne peut pas les copier ; avec les Superpositions d'états autorisées la mesure affecte l'état quantique [2]
Etant donné que les photons polarisés sont aléatoires, ce type d'échange est adapté à l'échange des clés mais pas pour l'échange de message puisque les photons polarisés sont générés aléatoirement alors qu'un message n'est pas aléatoire. Mais aussi une autre personne peut intercepter les photons. Si l'interception de la clé échangée a eu lieu, elle devrait être remarquée par les deux parties qui échangent la clé. La clé est alors rejetée et ils procèdent de nouveau à générer une autre clé. Si c'était un message qui est échangée, il aurait déjà perdu la confidentialité s'il est intercepté.

Le tableau 1 montre l'échange de la clé entre Alice et Bob en cryptographie quantique lorsqu'il y a présence de l'espion. Les lignes du tableau contiennent les bases utilisées tandis que les colonnes montrent les interlocuteurs qui échangent la clé. . Le tableau 2 montre l'échange de la clé en utilisant la cryptographie quantique entre Alice et Bob sans présence de l'espion. Les lignes du tableau contiennent les bases utilisées tandis que les colonnes montrent les interlocuteurs qui échangent la clé.

Tableau 1 Echange de clé sans espion

Aline			Bob			Alice et Bob
Base choisie	Symboles envoyés	Etat envoyé	Base choisie	Etat utilisé	Symbole mesuré	Clé secrète
x	0		+		1	
					0	
			x		0	0
					1	
+	1		x		1	
					0	
			+		0	
					1	1
x	0		+		1	
					0	
			x		0	0
					1	
+	1		x		0	
					1	1
			+		0	
					1	
x	1		+		0	
					1	
			x		0	
					1	1

Tableau 2 Echange de la clé avec présence de l'espion

Aline			Bob			Alice et Bob
Base choisie	Symboles envoyés	Etat envoyé	Base choisie	Etat utilisé	Symbole mesuré	Clé secrète
x	0		+		1	
					0	
			x		0	0
					1	
+	1		x		1	
					0	
			+		0	
					1	1
x	0		+		1	
					0	
			x		0	0
					1	
+	1		x		0	
					1	1
			+		0	
					1	
x	1		+		0	
					1	
			x		0	
					0	

II.6 .Cryptographie basée sur les courbes elliptiques.

II.6 .1.Généralités

Les courbes elliptiques ne sont pas des ellipses. Ils sont ainsi nommés car ils sont décrits par des équations cubiques, similaires à celles utilisées pour calculer la circonférence d'une ellipse. Les courbes elliptiques jouent un grand rôle en cryptographie. Du point de vue de l'échange de clé, la clé est échangée de manière secrète. En général, les équations cubiques pour les courbes elliptiques prennent la forme :$y^2+axy+by=x^3+cx^2+dx+e$ où a, b, c, d et e sont des nombres réels et x et y prennent des valeurs dans les nombres réels. La définition d'une courbe elliptique comprend également un seul élément noté O appelé le point à l'infini ou le point zéro. Deux familles de courbes elliptiques sont utilisées en cryptographie: courbes elliptiques sur Zp et courbes binaires sur GF(2^m).
L'équation est limitée à l'équation de la forme $y^2 = x^3 + ax + b$ pour des courbes elliptiques sur Zp. Pour les courbes elliptiques qui sont définies dans le corps binaire, l'équation est réduite à $y^2+xy= x^3 + ax^2 + b$

II.6 .2. Courbes elliptiques dans un corps binaire

Pour les courbes elliptiques qui sont définies dans un corps binaire, l'équation est réduite à $y^2+xy= x^3 + ax^2 + b$. les variables et les coefficients, éléments de GF(2^m) et les calculs sont effectués en GF (2^n).
Les opérations arithmétiques peuvent être effectuées dans la représentation binaire ou polynômiale .Pour la représentation polynômiale ; les opérations arithmétiques sont effectuées suivant les règles ordinaires de l'arithmétique polynômiale en utilisant les règles de base de l'algèbre, avec les deux améliorations suivantes :
- L'addition et la soustraction des coefficients sont effectuées modulo 2 :
1+1 =1-1 = 0; 1+0 = 1-0 = 1; 0+1 = 0-1 = 1
- Si la multiplication résulte en un polynôme de degré supérieur à n-1, le polynôme est réduit modulo certains polynômes irréductibles m(x) de degré n. C'est-à-dire qu'on divise par m(x) et garde le reste. Pour un polynôme f (x), le reste est exprimé comme r (x) = f (x) mod m(x).
Pour la représentation binaire, l'addition est effectuée en utilisant une opération logique XOR.

Exemple :

Soient $f(x)=x^6+x^4+x^2+x+1$, $g(x)=x^7+x+1$ *et* $m(x)=x^8+x^4+x^3+x+1$

$f(x)*g(x)=[f(x)*g(x)]\ mod\ m(x)= x^7+x^6+1$

La multiplication en utilisant la représentation binaire peut être définie comme suit : *Considérons un polynôme dans GF(2^8), qui a la forme* $f(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$. *Si on multiplie f(x) par x, on a* $x*f(x)= (b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x) \bmod m(x)$.*Le résultat* dépend du fait que le bit b_7 soit 0 ou 1.Si le bit b_7 de f(x) est égal à 0, on ne fait rien. Dans ce cas, le résultat est $b_6b_5b_4b_3b_2b_1b_00$.si b_7 est égal à 1, nous devons diviser le polynôme que nous avons pour f (x) * x par le polynôme de module m(x) et ne garder que le reste. Dans ce cas f(x)*x mod *m(x)* peut être écrite.

$(b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x) +(x^8 \bmod m(x))$.

$=(b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x) +(x^4 + x^3 + x + 1)$.

$b_6b_5b_4b_3b_2b_1b_00 \oplus 00011011$.

La multiplication est appliquée de facon suivante :

-Si on veut multiplier B_1 et B_2, chacun ayant une longueur de 8 bits.

-Si B_2 est 00000001, alors $B_1*B_{2=}$ B1. Le résultat est B1 lui-même.

-Si B_2 est 00000010, alors nous multiplions B_1 par X., la réponse dépend de la valeur $b_{7,}$ si b_7 est 0 le résultat est obtenu en faisant le décalage gauche d'un bit de $B_{1.}$ Si b_7 est 1,on fait d'abord le décalage gauche d'un bit de B_1 puis on fait le XOR du résultat obtenu avec 00011011 .
Alors f(x)*g(x)mod *m(x)* peut être résolu en binaire et aboutir au même résultat.
$f(x)=x^6 +x^4 +x^2 +x+1$ correspond 01010111
$g(x)=x^7 + x + 1$ correspond 10000011
(01010111)*(10000011)= (01010111)*(00000001+00000010+10000000)Tout d'abord, nous déterminons les résultats de la multiplication par des puissances de x.

01010111*00000001=01010111

01010111*00000010=10101110

01010111*00000100=01011100⊕00011011=01000111

01010111*00001000=10001110

01010111*00010000=00011100⊕00011011=00000111

01010111*00100000=00001110

01010111*01000000=00011100

01010111*10000000=00111000

(01010111)*(10000011)= (01010111)*(00000001+00000010+10000000)

= (01010111)*(00000001+00000010+10000000)

(01010111)*(00000001+00000010+10000000)

=01010111*00000001+01010111*00000010+01010111*10000000

=01010111+10101110+00111000

=01010111 ⊕ 10101110 ⊕ 00111000=11000001

$= x^7 + x^6 + 1$

II.6. 3. Addition de deux points dans GF(2^n)

Les règles d'addition des points P et Q peuvent être énoncées comme suit :

1.P+0=P

2. Si $P = (x_p, y_p)$, *alors* $P + (x_p, x_p + y_p) = O$.

3. Si $P = (x_p, y_p)$ *et* $Q = (x_q, y_q)$ *avec* $P \neq Q$,*alors* $R=P+Q= P = (x_R, y_R)$ *est déterminé par les règles suivants:*

$X_R=\lambda^2+\lambda+X_p+X_q+a$

$Y_R= \lambda(X_p+X_R)+X_R+Y_P$ *avec* $\lambda =(Y_q+Y_p)/(X_q+X_p)$

4.Si $P = (x_p, y_p)$,*alors* $R=2P= (x_R, y_R)$ *est déterminé par les règles suivants:*

$X_R=\lambda^2+\lambda+a$

$Y_R= X_p^2+ (\lambda +1)X_R$

avec $\lambda =X_p + X_p / Y_p$

II.6 .4. Addition de deux points de la courbe dans Zp

Pour les courbes elliptiques sur Zp,une équation cubique est utilisée dans laquelle les variables et les coefficients prennent des valeurs dans l'ensemble des nombres entiers de 0 à p-1 et dans lequel les calculs sont effectués mod p.

L'équation est décrite de la façon suivante :$y^2 \mod p = (x^3 + ax + b) \mod p$.

Dans ce point, sont décrites les propriétés permettant de calculer des additions sur des courbes elliptiques. Pour deux points $P = (x_P, y_P)$ et $Q = (x_q, y_q)$ qui ne sont pas négatifs l'un de l'autre. La somme R = P + Q peut être exprimée comme suit :

1er cas : si $p \neq q$

$\beta = (y_p - y_q) / (x_p - x_q)$

$x_r = \beta^2 - x_p - x_q$
$y_r = -y_p + \beta (x_p - x_r)$

[3]

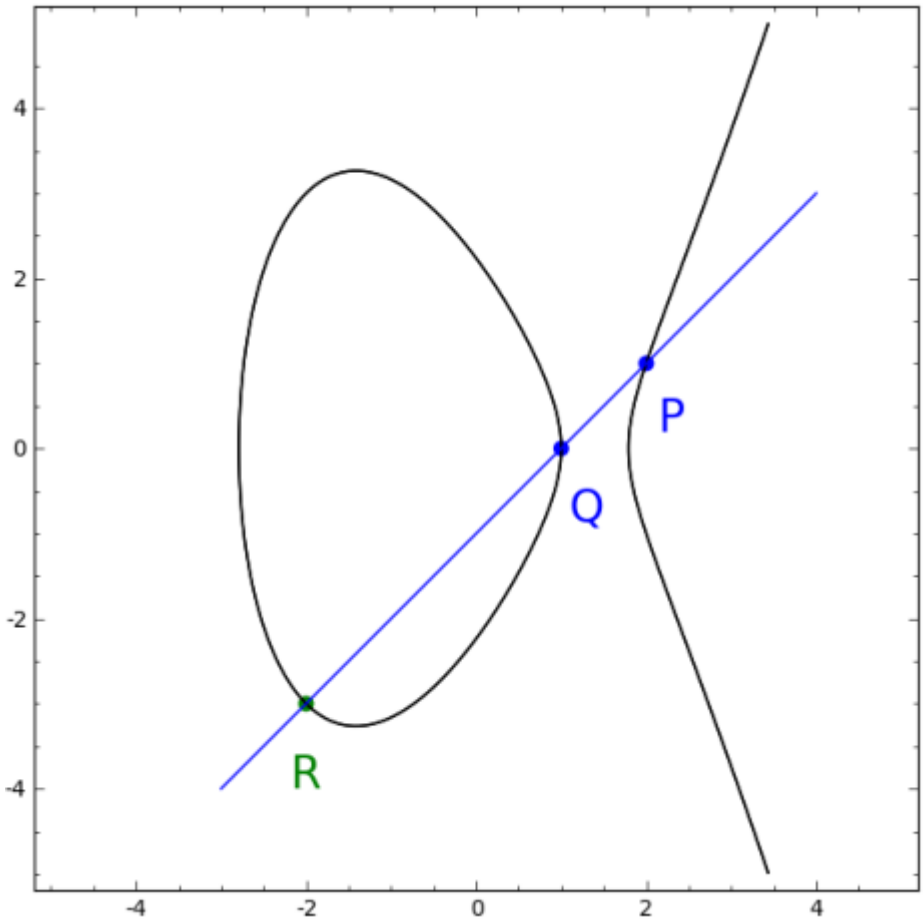


Figure 1 Addition de 2 points distincts

2eme cas : le doublement d'un point p.

$\beta = (3x_p^2 + a)/2y_p$, pour $y_p \neq 0$ et a est l'un des paramètres choisis avec la courbe elliptique .
$x_r = \beta^2 - 2x_p$
$y_r = -y_p + \beta (x_p - x_r)$,

[4]

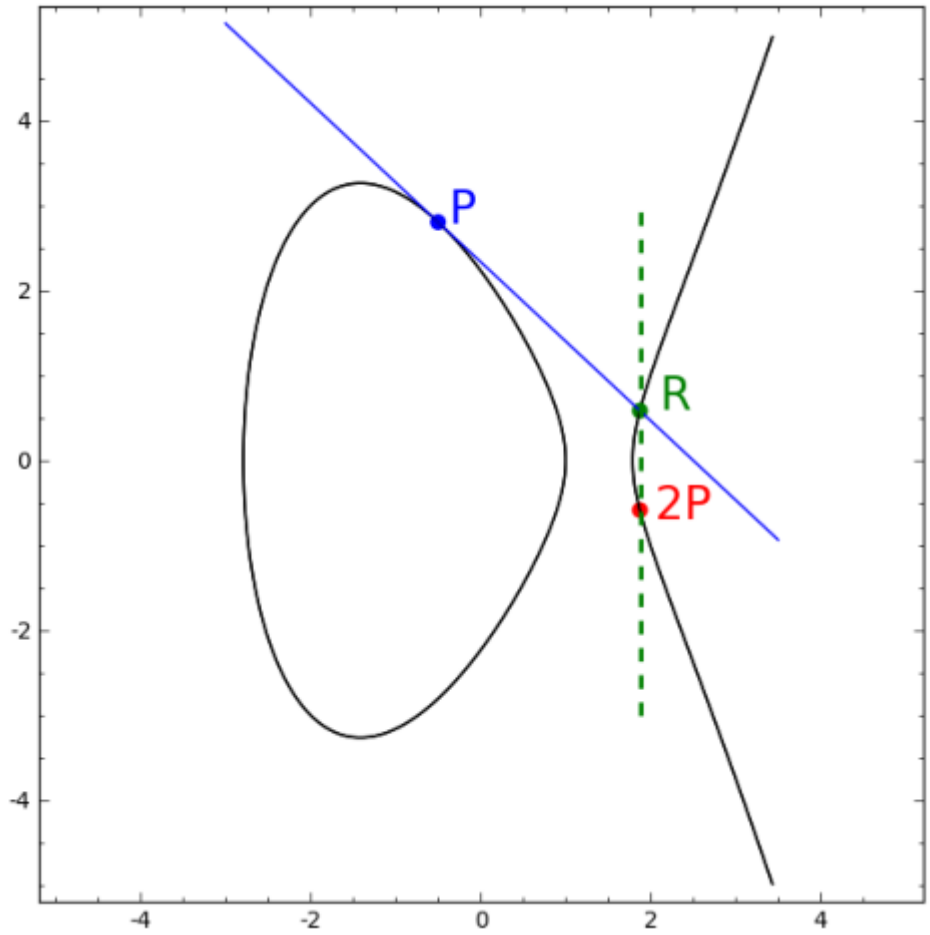


Figure 2 Doublement D'un point

p-p =0, correspond à $(x_1, y1) + (x_1, -y_1) = 0$, le point $(x_1, -y_1)$ est appelé le négatif de p.

II.6.5.Théorème de Hasse

Soit p un nombre premier et $q = p^k$. Soit E une courbe elliptique définie par
$E : y^2 = x^3 + ax + b$, avec $a,b \in Fq$.
Alors le nombre #E(Fq) de points de E(Fq) vérifie $|\#E(Fq) - (q + 1)| < 2\sqrt{q}$.
Soit E définie sur (Z/13Z) par $E : y^2 = x^3 - 5x + 8$.
Alors #E(F13) = 20 et on a bien $|\#E(F13) - (13 + 1)| = |20 - 14| = 6 < 2\sqrt{13}$.
[5]

II.6 .6. Echange de clé basée sur les courbes elliptiques

L'échange des clés à l'aide des courbes elliptiques peut être réalisé de la manière suivante : Choisissez d'abord un grand nombre entier q, qui est soit un nombre premier p soit un entier de la forme 2^n et les paramètres de la courbe elliptique a et b. Ceci définit le groupe de points elliptiques $E_q(a,b)$.Ensuite, choisissez une base, un point $G = (x_1, y_1)$ dans Eq (a, b) dont l'ordre est une très grande valeur n. L'ordre n d'un point G sur une courbe elliptique est le plus petit entier positif n tel que nG = O. $E_q(a,b)$ et G sont des paramètres connus par tous les deux interlocuteurs qui échangent la clé.

Un échange de clé entre les participants A et B peut être accompli comme suit :

1 A sélectionne un entier n_A inférieur à n. C'est la clé privée de A. A génère alors une clé publique $P_A = n_A * G$; la clé publique est un point dans Eq (a, b).
2. De même, B sélectionne une clé privée n_B et calcule une clé publique P_B
3. A génère la clé secrète $K = n_A * P_B$. B génère la clé secrète $K = n_B * P_A$.

Les éléments publics sont :
Eq (a, b) : La courbe elliptique avec les paramètres a,b et q ,où q est soit un nombre premier p soit un entier de la forme 2^n.
G : est un point sur la courbe elliptique dont l'ordre est une très grande valeur n .

La génération de la clé par l'utilisateur A:
A sélectionne un entier n_A avec $n_A < n$.
A calcule la valeur publique $PA = n_A * G$.
Calcul de la clé secrète par l'utilisateur A ;$K = n_A * P_B$.

La génération de la clé par l'utilisateur B:
B sélectionne un entier n_B avec $n_B < n$.
B calcule la valeur publique $P_B = n_B * G$.
Calcul de la clé secrète par l'utilisateur B ;$K = n_B * P_A$.
Les deux calculs produisent le même résultat car $n_A * P_B = n_A(n_B * G) = n_B(n_A * G) = n_B * P_A$.
Exemple d'illustration :
Soit la courbe elliptique $E_P(0,-4)$:$y^2 = x^3 - 4$,p=211.
G=(2,2) et 240G=0.n=240.
n_A=231
n_B=173
P_A=231(2,2)=(75,121)
PB=173(2,2)=(55,37)

La clé secrète K générée est 173(75,121)=231(55,37)=(124,92)

II.6 .7. Algorithme de cryptage et de décryptage

Pour chiffrer et envoyer un message P_m à B, A choisit un entier positif aléatoire k et produit le chiffré C_m composé de la paire des points:
$C_m=(kG, P_m +kP_B)$.
G doit être choisi parmi les points de la courbe elliptique.

Pour $E_{211}(2,3)$ les points de cette courbe sont :
(1,46)(1,165)(3,6)(3,205)(6,81)(6,130)(8,98)(8,113)(9,31)(9,180)(10,50)
(10,161)(16,79)(16,132)(19,88)(19,123)(20,5)(20,206)(23,20)(23,191)(25,8)
(25,203)(27,31)(27,180)(29,94)(29,117)(30,104)(30,107)(31,59)(31,152)(33,6
6)(33,145)(36,10)(36,201)(39,98)(39,113)(40,19)(40,192)(41,3)(41,208)(42,89
)
(42,122)(43,7)(43,204)(45,53)(45,158)(51,44)(51,167)(52,20)(52,191)(53,81)
(53,130)(54,13)(54,198)(55,3)(55,208)(56,50)(56,161)(57,91)(57,120)(58,30)
(58,181)(60,103)(60,108)(61,80)(61,131)(62,5)(62,206)(65,6)(65,205)(67,15)
(67,196)(69,40)(69,171)(72,66)(72,145)(73,90)(73,121)(74,39)(74,172)(78,52)
(78,159)(79,84)(79,127)(80,53)(80,158)(84,78)(84,133)(86,53)(86,158)(88,11)
(88,200)(89,58)(89,153)(90,38)(90,173)(91,8)(91,203)(95,8)(95,203)(96,29)
(96,182)(97,57)(97,154)(98,72)(98,139)(101,68)(101,143)(105,9)(105,202)
(106,66)(106,145)(107,48)(107,163)(110,92)(110,119)(111,24)(111,187)(115,
3)(115,208)(119,96)(119,115)(123,27)(123,184)(125,28)(125,183)(128,55)
(128,156)(129,5)(129,206)(130,72)(130,139)(131,28)(131,183)(133,16)
(133,195)(136,20)(136,191)(137,54)(137,157)(139,9)(139,202)(143,6)(143,20
5)
(144,25)(144,186)(145,50)(145,161)(147,2)(147,209)(148,42)(148,169)(152,8
1)
(152,130)(153,43)(153,168)(156,29)(156,182)(160,32)(160,179)(164,98)
(164,113)(166,28)(166,183)(169,37)(169,174)(170,29)(170,182)(173,45)
(173,166)(175,31)(175,180)(178,9)(178,202)(179,64)(179,147)(183,19)
(183,192)(184,10)(184,201)(185,47)(185,164)(192,51)(192,160)(193,26)
(193,185)(194,72)(194,139)(195,99)(195,112)(196,94)(196,117)(197,94)
(197,117)(198,34)(198,177)(199,19)(199,192)(202,10)(202,201)(206,76)
(206,135)(210,0) , avec le point à l'infini.

Pour $E_{751}(-1,188)$ les points de cette courbe sont :

(0,375)(0,376)(1,375)(1,376)(2,373)(2,378)(3,211)(3,540)(5,225)(5,526)(6,361
)(6,390)(7,240)(7,511)(12,235)(12,516)(13,129)(13,622)(17,332)(17,419)
(18,137)(18,614)(19,138)(19,613)(21,133)(21,618)(24,65)(24,686)(26,312)
(26,439)(28,302)(28,449)(30,236)(30,515)(32,300)(32,451)(34,118)(34,633)

(36,336)(36,415)(38,256)(38,495)(39,349)(39,402)(41,274)(41,477)(43,322)
(43,429)(44,312)(44,439)(45,97)(45,654)(47,335)(47,416)(50,136)(50,615)
(52,346)(52,405)(54,131)(54,620)(57,332)(57,419)(59,325)(59,426)(62,161)
(62,590)(63,62)(63,689)(64,13)(64,738)(66,214)(66,537)(67,22)(67,729)(69,2
1)(69,730)(72,285)(72,466)(74,199)(74,552)(77,6)(77,745)(79,333)(79,418)
(84,138)(84,613)(86,184)(86,567)(92,145)(92,606)(93,120)(93,631)(94,73)(94
,678)(97,284)(97,467)(101,4)(101,747)(102,120)(102,631)(103,180)(103,571)(
121,39)(121,712)(124,354)(124,397)(126,275)(126,476)(128,252)(128,499)(1
31,34)(131,717)(135,198)(135,553)(139,338)(139,413)(144,190)(144,561)(14
7,175)(147,576)(152,337)(152,414)(153,315)(153,436)(154,176)(154,575)(15
5,193)(155,558)(157,55)(157,696)(158,29)(158,722)(159,124)(159,627)(160,1
40)(160,611)(161,1)(161,750)(169,132)(169,619)(170,274)(170,477)(172,255)
(172,496)(173,33)(173,718)(174,240)(174,511)(177,164)(177,587)(178,232)(1
78,519)(179,257)(179,494)(180,343)(180,408)(182,84)(182,667)(187,50)(187,
701)(188,94)(188,657)(189,23)(189,728)(190,196)(190,555)(191,29)(191,722)
(192,359)(192,392)(195,57)(195,694)(196,113)(196,638)(197,107)(197,644)(1
98,322)(198,429)(200,220)(200,531)(201,5)(201,746)(202,13)(202,738)(203,1
69)(203,582)(204,363)(204,388)(205,161)(205,590)(207,215)(207,536)(209,1
98)(209,553)(210,97)(210,654)(211,20)(211,731)(212,250)(212,501)(214,76)(
214,675)(217,247)(217,504)(219,38)(219,713)(224,248)(224,503)(225,117)(2
25,634)(227,74)(227,677)(229,115)(229,636)(231,176)(231,575)(232,50)(232,
701)(233,256)(233,495)(234,188)(234,563)(237,23)(237,728)(238,55)(238,69
6)(239,374)(239,377)(241,230)(241,521)(243,336)(243,415)(245,242)(245,50
9)(246,53)(246,698)(248,236)(248,515)(251,56)(251,695)(253,108)(253,643)(
254,247)(254,504)(256,342)(256,409)(257,149)(257,602)(258,276)(258,475)(
261,288)(261,463)(262,208)(262,543)(263,354)(263,397)(264,337)(264,414)(
265,76)(265,675)(266,244)(266,507)(267,352)(267,399)(268,229)(268,522)(2
69,224)(269,527)(270,162)(270,589)(272,76)(272,675)(274,329)(274,422)(27
8,241)(278,510)(279,83)(279,668)(280,247)(280,504)(283,54)(283,697)(285,9
6)(285,655)(291,16)(291,735)(295,110)(295,641)(296,245)(296,506)(297,182)
(297,569)(299,183)(299,568)(301,258)(301,493)(302,291)(302,460)(304,165)(
304,586)(305,268)(305,483)(310,31)(310,720)(311,111)(311,640)(312,95)(31
2,656)(314,36)(314,715)(317,181)(317,570)(318,79)(318,672)(320,174)(320,5
77)(324,7)(324,744)(325,23)(325,728)(329,369)(329,382)(331,367)(331,384)(
332,50)(332,701)(333,316)(333,435)(335,337)(335,414)(337,178)(337,573)(3
39,353)(339,398)(341,362)(341,389)(343,205)(343,546)(348,228)(348,523)(3
52,65)(352,686)(354,153)(354,598)(356,55)(356,696)(358,200)(358,551)(359,
296)(359,455)(361,8)(361,743)(364,354)(364,397)(366,176)(366,575)(367,12
2)(367,629)(370,11)(370,740)(372,37)(372,714)(373,9)(373,742)(375,65)(375,
686)(378,172)(378,579)(380,155)(380,596)(381,69)(381,682)(384,306)(384,4
45)(385,328)(385,423)(386,72)(386,679)(387,59)(387,692)(389,146)(389,605)
(391,187)(391,564)(392,341)(392,410)(393,41)(393,710)(394,324)(394,427)(3
95,141)(395,610)(398,326)(398,425)(400,238)(400,513)(401,153)(401,598)(4

02,29)(402,722)(403,218)(403,533)(405,92)(405,659)(406,194)(406,557)(407,198)(407,553)(409,281)(409,470)(412,191)(412,560)(414,88)(414,663)(416,371)(416,380)(417,158)(417,593)(421,362)(421,389)(423,135)(423,616)(426,201)(426,550)(430,102)(430,649)(432,310)(432,441)(440,294)(440,457)(446,219)(446,532)(447,303)(447,448)(452,278)(452,473)(454,280)(454,471)(461,92)(461,659)(465,181)(465,570)(466,143)(466,608)(467,249)(467,502)(470,310)(470,441)(472,336)(472,415)(473,236)(473,515)(479,355)(479,396)(480,256)(480,495)(484,161)(484,590)(485,13)(485,738)(486,355)(486,396)(487,371)(487,380)(488,59)(488,692)(490,207)(490,544)(492,167)(492,584)(493,48)(493,703)(495,356)(495,395)(496,97)(496,654)(500,368)(500,383)(501,155)(501,596)(503,241)(503,510)(508,83)(508,668)(510,322)(510,429)(512,298)(512,453)(513,141)(513,610)(514,201)(514,550)(520,328)(520,423)(522,282)(522,469)(529,254)(529,497)(531,194)(531,557)(532,44)(532,707)(533,74)(533,677)(537,355)(537,396)(538,77)(538,674)(540,274)(540,477)(543,61)(543,690)(545,370)(545,381)(546,352)(546,399)(547,299)(547,452)(549,244)(549,507)(551,231)(551,520)(553,230)(553,521)(555,307)(555,444)(556,120)(556,631)(562,201)(562,550)(565,194)(565,557)(566,344)(566,407)(570,240)(570,511)(575,329)(575,422)(578,99)(578,652)(579,286)(579,465)(581,20)(581,731)(582,15)(582,736)(583,177)(583,574)(584,271)(584,480)(585,108)(585,643)(586,162)(586,589)(589,79)(589,672)(591,104)(591,647)(592,91)(592,660)(594,141)(594,610)(595,79)(595,672)(597,328)(597,423)(599,371)(599,380)(600,310)(600,441)(603,164)(603,587)(604,210)(604,541)(605,297)(605,454)(607,18)(607,733)(609,125)(609,626)(612,163)(612,588)(617,149)(617,602)(620,106)(620,645)(621,155)(621,596)(622,242)(622,509)(623,47)(623,704)(624,295)(624,456)(627,59)(627,692)(628,149)(628,602)(629,206)(629,545)(632,80)(632,671)(634,94)(634,657)(635,242)(635,509)(636,92)(636,659)(637,301)(637,450)(639,343)(639,408)(646,162)(646,589)(648,138)(648,613)(649,57)(649,694)(650,58)(650,693)(653,329)(653,422)(654,263)(654,488)(657,156)(657,595)(658,57)(658,694)(660,257)(660,494)(661,179)(661,572)(663,257)(663,494)(664,108)(664,643)(666,85)(666,666)(667,180)(667,571)(671,193)(671,558)(672,212)(672,539)(676,193)(676,558)(677,332)(677,419)(678,42)(678,709)(680,94)(680,657)(681,312)(681,439)(683,343)(683,408)(684,119)(684,632)(686,358)(686,393)(687,244)(687,507)(688,327)(688,424)(689,352)(689,399)(693,70)(693,681)(694,199)(694,552)(695,170)(695,581)(696,90)(696,661)(697,279)(697,472)(701,203)(701,548)(704,186)(704,565)(705,112)(705,639)(708,230)(708,521)(710,20)(710,731)(712,100)(712,651)(714,173)(714,578)(715,83)(715,668)(717,150)(717,601)(720,181)(720,570)(721,241)(721,510)(722,164)(722,587)(723,304)(723,447)(729,78)(729,673)(731,222)(731,529)(732,180)(732,571)(733,265)(733,486)(734,199)(734,552)(735,46)(735,705)(740,362)(740,389)(742,74)(742,677)(743,154)(743,597)(745,27)(745,724)(747,153)(747,598)(750,375)(750,376) avec le point à l’infini.

Prenons le point (0,376) comme un générateur G. prenons le point P_m à crypter est (443,253) .pour crypter ce point l'interlocuteur A utilise la clé publique de l'interlocuteur B.B ayant choisi n_B=85
$P_B=n_BG$=85(0,376)=(671,558) .L'interlocuteur A choisit k=113.
Pour crypter le point p_m, il calcule
$=[(kG),(p_m+kp_B)]=[113*(0,376),(443,253)+113*(671,558)$
$=[(34,633),(443,253)+(47,416)]$
$=[(34,633),(217,606)]$

Pour décrypter ,on calcule $(P_m+kP_B)-[n_B(kG)]$
$(P_m+kP_B)-[n_B(kG)]=(217,606)-[85(34,633)]$
$=(217,606)-[(47,416)]$
$=(217,606)+[(47,-416)]$
$=(217,606)+[(47,335)]$
$=(443,253)$

L'importance de cet algorithme est basée sur la difficulté de résoudre le problème du logarithme discret des courbes elliptiques. En effet, il est difficile de déterminer k à partir de kp et p.

II.6.8 Algorithme de signature numérique

Pour utiliser cet algorithme de signature numérique à clé publique utilisant les courbes elliptiques communément appelé ECDSA, les deux parties doivent se mettre d'accord sur un ensemble d'éléments qui définissent la courbe elliptique (p,Ep(a,b),G, n). Chaque participant génère ses paramètres de domaine et les envoie avec sa clé publique.

Génération des clés:

Chaque entité A doit faire ce qui suit:
1. Sélectionnez une courbe elliptique E définie sur Zp. Le nombre de points dans E (Zp) doit être divisible par un grand nombre premier n.
2. Sélectionnez un point P ∈ E (Zp) d'ordre n.
3. Sélectionnez un entier d statistiquement unique et imprévisible dans l'intervalle [1 ,n-1]
4. Calculez Q = dP.
5. Une clé publique est (E, P, n, Q); la clé privée de A est d.

génération de signature ECDSA:

Pour signer un message m, A doit procéder comme suit:
1. Sélectionnez un entier k aléatoire dans l'intervalle [1, n - 1].
2. Calculez kP = (x_1, y_1) et r = x1 mod n. Si r = 0, retournez à l'étape 1.

3. Calculez k^{-1} mod n.
4. Calculer $s = k^{-1}(h(m) + dr)$ mod n,où h(m) est l'algorithme de hachage sécurisé.
5. Si s = 0, retournez à l'étape 1. (Si s = 0, alors s^{-1} mod n n'existe pas; s^{-1} est utilisé à l'étape 3 de la vérification de la signature.)
6. La signature du message m est la paire d'entiers (r, s).

• *Vérification de la signature ECDSA.*

Pour vérifier la signature (r, s) de A sur le message m, B doit procéder comme suit:
1. Obtenez une copie certifiée conforme de la clé publique (E, P, n, Q) de A.
2. Vérifiez que r et s sont des entiers dans l'intervalle [1, n - 1].
3. Calculez $w = s^{-1}$ mod n et h (m).
4. Calculez $u_1 = h(m)\, w$ mod n et $u_2 = rw$ mod n.
5. Calculez $R = u_1P + u_2Q = (x_0, y_0)$ et $v = x_0$ mod n. Si R = O, renvoyez «invalide» et arrêtez.
6. Acceptez la signature si et seulement si v = r. Si v ≠ r retourne invalide.

Exemple d'illustration
E :$y^2 = x^3 + 2x + 6$mod 7
La courbe elliptique E a 11 points, donc n =11,k=6,m= hello class, h(m)= 1775530154
P=(1,3)
D=4
Q=dP= (3,5)
kP=(4,6)
r=4mod 11=4
k^{-1}mod 11=2
$s = k^{-1}(m+dr)$mod 11=3
La signature est le couple (4,3)
Vérification de la signature:
$w = s^{-1}$ mod n =3^{-1}mod11=4
$u_1 = m\, w$ mod n =8
$u_2 = rw$ mod n=5
u_1P= (5,6)
u_2Q= (2,5)
$R = u_1P + u_2Q = (x_0, y_0)$ =(4,6)
$v = x_0$ mod n.=4mod 11=4
r=v, la signature est validée

II.6 .9.Algorithmes résolvant le problème du logarithme discret

Avec la recherche exhaustive. on calcule kp=Q, pour k allant de 2 jusqu' à trouver la valeur recherchée de Q.

Exemple d'illustration.
Prenons l'exemple dans $E_{23}(9,17)$ qui a comme points :

(1,2)(1,21)(3,5)(3,18)(4,5)(4,18)(5,7)(5,16)(7,3)(7,20)(8,7)(8,16)(10,7)(10,16)(12,6)(12,17)(13,10)(13,13)(14,9)(14,14)(15,10)(15,13)(16,5)(16,18)(17,0)(18,10)(18,13)(19,3)(19,20)(20,3)(20,20) avec le point à l'infini.

Calculons le logarithme discret k de Q=(3,18) à la base P=(16,5)
P = (16, 5); 2P = (20, 20); 3P = (14, 14); 4P = (19, 20); 5P = (13, 10); 6P = (7, 3); 7P = (8, 7); 8P =(12, 17); 9P = (4, 5), 10P= (3,18).
Comme 10p=(3,18)=Q, le logarithme discret de Q=(3,18) à la base p=(16 ,5) est k qui est égale à 10.
Il est alors très difficile de résoudre un tel problème pendant un temps raisonnable pour les grands nombres premiers.

Avec la méthode de pas de bébé et pas de géant on procède comme suit :

Ayant un point p sur la courbe elliptique, et n étant l'ordre de la courbe elliptique. on calcule $m \geq \sqrt{n}$ tel que $p+1-2p^{1/2} \leq m \leq p+1+2p^{1/2}$ et mp=0.L'algorithme de la méthode de pas de bébé et pas de géant est le suivant :Q est un générateur d'ordre n et un point donné A.tel que kQ=A.On construit deux listes ,la première liste contient jmq pour chaque j , $0 \leq j \leq m-1$. La 2^{eme} liste contient -iq+A pour chaque i, $0 \leq i \leq m-1$.On s'arrête si on trouve un même point dans les deux listes.

P=jmQ

P=-iQ+A

jmQ=-iQ+A

(jm+i)Q=A.Le multiple de Q tel kQ=A est k=jm+i. A doit être un multiple de Q pour que cet algorithme fonctionne.

Exemple :soit la courbe elliptique $E_{23}(0,4)$ ou $y^2 \bmod 23=(x^3+4) \bmod 23$, l'ordre de cette courbe est 29 ,m=6 ,Q=(0,2) est le générateur et le point (18,9) à déterminer k tel que kQ=(18,9). Dans le tableau 3 on détermine les valeurs de jmQ pour les valeurs de j.

Tableau 3 Pas de géant pour les courbes elliptiques

j	1	2	3	4	5
jmQ	(9,11)	(17,9)	(10,18)	(7,3)	

Dans le tableau 3 on détermine les valeurs de iQ+A pour les valeurs de i.

Tableau 4 Pas de bébé les courbes elliptiques

i	1	2	3	4	5
iQ+A	(8,15)	(17,9)			

Comme l'élément (17,9) de la deuxième liste correspond à l'élément (17,9) de la première liste, (17,9) est considéré pour les deux listes. En appliquant les égalités suivantes, k est trouvé :

12Q=-2Q+A ,14Q=A ,K=jm+i=14

II.10. Conclusion

Les cryptosystèmes pour lesquels l'échange de clé est sécurisé sont les meilleurs cryptosystèmes puisque la clé est échangée de manière à éviter qu'une personne qui n'est pas concernée par la conversation ne connaisse pas la clé. Mais il faut obligatoirement contourner l'attaque dite l'homme du milieu en utilisant la signature numérique pour le cryptosystème de Diffie-Hellman, le cryptosystème El Gamal et le cryptosystème basé sur les courbes elliptiques.

Comme le cryptosystème de Diffie-Hellman permet de générer une clé qui est utilisée en cryptographie symétrique, dans notre projet nous avons choisi le cryptosystème de Diffie-Hellman pour utiliser la clé générée par ce cryptosystème dans des cryptosystèmes symétriques pour profiter de la rapidité de la cryptographie symétrique. On aurait choisi la cryptographie quantique mais les outils pour la mise en application de ce cryptosystème sont chers.

CHAPITRE III IMPLEMENTATION DU CRYPTOSYSTEME DE DIFFIE HELLMAN

III.1. Introduction

L'homme a toujours voulu cacher ce qu'il communique à des personnes qui ne sont pas destinées à avoir le message communiqué. Le cryptosystème Diffie Hellman ne consiste pas à chiffrer les données mais à générer la même clé cryptographique aux deux extrémités des interlocuteurs, de sorte qu'il n'est pas nécessaire de transférer cette clé à une autre personne qu'on veut communiquer les données en utilisant la cryptographie symétrique puisque la même clé est calculée aux deux extrémités de deux interlocuteurs. Il s'agit de s'entendre sur une clé que deux parties peuvent utiliser pour le cryptage symétrique, de telle manière qu'une autre personne en dehors de ces deux parties ne puisse pas obtenir la clé. L'importance du cryptosystème Diffie Hellman étant basée sur la difficulté de la résolution du problème du logarithme discret, un tel problème peut être résolu pour les petits nombres premiers. La signature numérique est indispensable pour garantir l'intégrité des données et l'authenticité et la non-répudiation de l'expéditeur des données.

III.2. But et fonctionnement de l'algorithme Diffie-Hellman

L'algorithme de Diffie-Hellman permet aux personnes de se mettre d'accord sur un secret commun en échangeant des messages publiquement qu'un espion ne peut pas intercepter. Les deux personnes qui veulent communiquer utilisent les nombres p,et g qui sont publics. Considérons l'échange des clés entre 2 personnes, l'une appelée A et l'autre appelée B. La personne A choisit un nombre privé a et envoie à B le résultat de (g^amod p). B choisit un nombre privé b et envoie à A le résultat de (g^bmod p).
A calcule $Z_A=(g^b \text{mod} p)^a \text{mod} p=g^{ba} \text{mod} p$.
B calcule $Z_B=(g^a \text{mod} p)^b \text{mod} p=g^{ab} \text{mod} p$.ZA=ZB est la clé privée pour les deux personnes.

La sécurité de l'algorithme Diffie-Hellman réside dans la difficulté de trouver ab ou ba comme exposant de g pour ensuite calculer ZA=ZB qui est la clé privée pour les deux personnes. En d'autre termes c'est très difficile de trouver x tel que $\log_a b=x$ ou $a^x=b$, dans quelques groupes comme le groupe multiplicatif et surtout voire presque impossible pour des grands nombres premiers. Ce problème est appelé, le problème du logarithme discret.

III.3.Problème du logarithme discret

III.3.1. Problème du logarithme discret dans le group Z_p^*

Soit l'ensemble $Z_P^* = \{1,2,3,4...,p-1\}$ où P est un nombre premier différent de 2.$(Z_P^*, .)$ est un group où toutes les propriétés pour qu'il soit group sont vérifiées.Soient $\beta, \alpha \in Z_P^*$, $\alpha . \beta = (\alpha \beta \bmod p)$

III.3.2.Quelques propriétés de Z_P^*

1.Si $\alpha, \beta \in Z_P^*$ alors $\alpha.\beta \in Z_P^*$,elle est interne et partout définie dans $\boldsymbol{Z_P^*}$

2.Si $\alpha, \beta, \lambda \in Z_P^*$ alors $(\alpha. \beta).\lambda = \alpha. (B.\lambda)$ elle est associative

3. Si $\alpha \in Z_P^*$ alors 1. $\alpha = \alpha$, elle admet 1 comme élément neutre.

4. $\alpha \in Z_P^*$,alors il existe $\beta \in Z_P^*$ tel que $\alpha.\beta = 1$.,Tout élément de Z_P^* a son élément symétrique.

Considérons P=17,alors $\boldsymbol{Z_{17}^*} = \{1,2,3,4...,16\}$

Propriété 1 :$5.6 = 30 \bmod 17 = 13$ alors $5.6 \in \boldsymbol{Z_{17}^*}$ .La propriété est vérifiée.

Propriété 2: $5,6,4 \in \boldsymbol{Z_{17}^*}$ vérifions si $(5.6).4 = 5.(6.4)$

$$(5.6).4 = (30 \bmod 17).4 = 13.4 = 52 \bmod 17 = 1$$

$$5.(6.4) = 5.(24 \bmod 17) = 5.7 = 35 \bmod 17 = 1$$

La propriété est vérifiée.

Propriété 3 : $14 \in Z_{17}^*$,$1.14 = 14 \bmod 17 = 14$

Propriété 4: $3 \in Z_{17}^*$,il existe $\beta \in Z_P^*$,tel que $3.\beta = 1$

$$3\beta \bmod 17 = 1, 3\beta \equiv 1 \bmod 17$$

$(Z_P^*, .)$ est un groupe particulier appelé groupe cyclique. Un groupe cyclique est un groupe dont au moins un des ses éléments est un générateur.

Soit $\alpha \in Z_P^*$,α est un générateur si $\{\alpha, \alpha^2, \alpha, \ldots, \alpha^{p-1}\} = Z_P^*$ .En d autres termes tous les éléments de Z_P^* sont obtenus à partir de α exposant les valeurs de l'ensemble $\{1,2,3,\ldots, p-1\}$ et α comme base .$\alpha \in Z_P^*$est un générateur si et seulement si $\alpha^{(p-1)/q} \bmod p \neq 1$,pour tout nombre premier q tel que p-1 est divisible par q.

Z_P^* étant { α, α^2, α^3 ,…, α^{p-1}} ,pour tout α générateur de Z_P^*d'après le petit théorème de Fermat , $\alpha^{p-1}=1$. Z_P^* devient { 1,α, α^2, α^3 ,…, α^{p-2}}.si $\beta \in Z_P^*$,alors $\beta= \alpha^x$,$0\leq x \leq p-2$,x est appelé logarithme discret de β de base α.$Log_\alpha\beta=x$ ou $\beta= \alpha^x$ en Z_P^*,$0\leq \log_\alpha \beta \leq p-2$.

Le problème du logarithme discret est de trouver $x \in \{0,1,2,\ldots,p-2\}$ tel que $\alpha^x \equiv \beta \bmod p$, ou trouver $Log_\alpha\beta$. P étant un nombre premier supérieur à 2, α un générateur de Z_P^*, $\beta \in Z_P^*=\{1,2,\ldots,p-1\}$.Il est alors très difficile voire impossible de calculer $Log_\alpha\beta$ dans Z_P^* pour le grand nombre premier p. La difficulté de calculer $\log_\alpha\beta$ ne dépend pas du générateur α mais dépend surtout du nombre premier p, plus le nombre premier est assez plus grand ,plus il est difficile de calculer $\log_\alpha\beta$.La recherche exhaustive est une approche naïve utilisée pour trouver $\log_\alpha\beta$.les calculs α^1, α^2, α^3 ,…, α^{p-1} sont faits jusqu' à ce que β soit trouvé .Mais il est clairement remarqué que cette méthode ne peut aboutir au résultat à un temps raisonnable pour le grand nombre premier.

III.3.3.Problème du logarithme discret dans le groupe Z_n^*

Z_n^* est défini comme l'ensemble des nombres naturels appartenant dans {1,…,n} qui sont premiers avec n, pour tout $n \geq 1$. (Z_n^*,.) est un groupe cyclique si au moins l'une des propriétés suivantes est vérifiée :

1)n=2 ou n= 4

2)$n=p^k$;$k \in \{1,2,3,\ldots\}$ p est un nombre premier.

3) $n=2.p^k$;$k \in \{1,2,3,\ldots\}$ p est un nombre premier.

$\alpha \in Z_n^*$ est un générateur si est seulement si $\alpha^{\varphi(n)/q} \bmod n \neq 1$, pour tout nombre premier q tel que φ(n) est divisible par q.Pour (Z_n^*,.) qui est un groupe cyclique, il existe φ (φ(n)) générateurs.

Le problème du logarithme discret dans Z_n^* est de calculer $\log_\alpha\beta$, autrement dit, trouver x pour l'équation $\beta= \alpha^x$ tel que $0\leq x \leq \varphi(n)-1$ pour $n=2,4,p^k.2p^k$ avec $k \in \{1,2,3,\ldots\}$ et p un nombre premier impair, α un générateur de Z_n^*, $\beta \in Z_n^*$.

$Z_n^*= \{ \alpha^o, \alpha^1, \alpha^2, \ldots, \alpha^{\varphi(n)-1}\}$

III.3.4. Quelques algorithmes utilisés pour résoudre le problème du logarithme discret.

Le problème est de déterminer x tel que $Log_{\alpha}\beta=x$ ou $\beta= \alpha^{x}$. L'algorithme de la recherche exhaustive et l'algorithme de pas de bébé et pas de géant de Shanks sont traités dans notre travail. Ces deux algorithmes peuvent être utilisés dans les deux groupes traités.

Algorithme de la recherche exhaustive consiste à calculer les valeurs suivantes α^{1}, α^{2} ,…, $\alpha^{\varphi(n)-1}$ jusqu' à obtenir x , α et β étant connus d'avance. Il est évident que cet algorithme n'est pas efficace pour les grands nombres.

Pour l'algorithme de pas de bébé et pas de géant de Shanks il s'agit toujours de calculer x tel que $Log_{\alpha}\beta=x$ ou $\beta= \alpha^{x}$.Pour cet algorithme de pas de bébé et pas de géant de Shanks, $m= \lceil\sqrt{\varphi(n)}\rceil$ est d'abord calculé et x est exprimé de la façon suivante :$x=x_g.m+x_{b.}$, $0\leq b_x,g_x < m$. L'équation $\beta= \alpha^{x}$ devient $\beta= \alpha^{xg.m+xb}$.

$\beta= \alpha^{xg.m}. \alpha^{xb}$

$\alpha^{xb}= \beta /\alpha^{xg.m}$

$\alpha^{xb}= \beta\, \alpha^{-m.xg}$

Les valeurs b_x et g_x sont déterminées en passant par la méthode de pas de bébé et pas de géant. La méthode de pas de bébé est utilisée en premier lieu. Pour cette méthode on calcule les valeurs α^{xb} tel que $0\leq \alpha^{xb} < m$.ces valeurs sont mémorisées pour l'utilisation ultérieure. Deuxièmement la méthode pas de géant est utilisée, elle consiste à déterminer le premier x_g ;$0\leq x_g < m$,tel que $\alpha^{xb}= \beta.(\alpha^{-m.})^{xg}$ .Si l'équation $\alpha^{xb}= \beta.(\alpha^{-m.})^{xg}$ est évaluée pour la valeur de x_g et $x_{b.,}$l'algorithme est arrêté et le x est déterminé par expression $x=x_g.m+x_{b\,..}$L'exemple suivant est résolu à titre d'exemple en utilisant l'algorithme de pas de bébé/pas de géant de Shank .

On détermine x, dans le cas suivant $\log_3 57 =x$ ou $3^{x}=57$ en $z^*_{113.}$

$m= \lceil\sqrt{\varphi(113)}\rceil=11.$

La méthode de pas de bébé est appliquée et les valeurs trouvées sont dans le tableau suivant.

Tableau 5 pas de bébé pour le cryptosystème Diffie-Hellman

x_b	0	1	2	3	4	5	6	7	8	9	10
$\alpha^{xb} \bmod 113$	1	3	9	27	81	17	51	40	7	21	63

La méthode de pas de géant est appliquée en utilisant l'expression $57.(3^{-1})^{11.xg}$,pour tout x_g tel que $0 \leq xg < m$. on détermine l'inverse de 3 en utilisant l'algorithme d'Euclide étendu.Si $\alpha \in Z_n^*$,PDCD(α,n)=1.s et t sont déterminés tel que $\alpha.s+t.n=1$.Alors $\alpha^{-1}=s \bmod n$.

113=1.113+0.3

3=0.113+1.3

2=1+113-37.3

1=-1.113+38.3

Alors s=38, $3^{-1}=38 mod\ 113=38$

M étant 11, $\alpha^{-m}=38^{11} mod\ 113=58$

Les valeurs du pas de géant sont dans la table suivante : pour x_g défini dans l'intervalle de 0 à11, 11 exclus.

Tableau 6 Pas de géant pour le cryptosystème Diffie-Hellman

x_g	0	1	2	3	4	5	6	7	8	9	10
$57.(3^{-1})^{11.xg}$	57	29	100	37	112	55	26	39	2	3	61

On remarque que 3 est dans le tableau de pas de bébé et de pas de géant. x_b et x_g sont respectivement 1 et 9 pour cette valeur 3.alors X=11*9+1=100.La vérification peut être faite et donne le même résultat :$3^{100} \bmod 113=57$.L'algorithme de pas de bébé et pas de géant de Shanks exige une mémoire vaste pour stocker les valeurs .Pour des grands nombres, cette méthode n'est pas efficace.

III.4. Diffie-Hellman et l'attaque de l'homme du milieu

Bien que l''échange de la clé de Diffie-hellman présente un avantage basé sur le problème du logarithme discret, il peut être soumis à l'attaque de

l'homme du milieu ou man in the middle. Si les deux personnes A et B veulent échanger des valeurs publiques nécessaires pour générer des clés. Pour une attaque dite man in the middle, une autre personne C, intercepte la valeur publique de A et envoie sa propre valeur publique à B: lorsque B transmet sa valeur publique, C la substitue alors C et A se mettent donc d'accord sur une clé partagée et C et B se mettent d'accord sur une autre clé partagée. Après cet échange, C décrypte simplement tous les messages envoyés par A ou B, puis lit et éventuellement modifie avant de recrypter avec la clé appropriée et de les transmettre à l'autre partie .Cette vulnérabilité est présente car l'échange de clé Diffie-Hellman n'authentifie pas les participants. Dans notre travail ce problème est résolu par la signature numérique.

III.5. Signature numérique

Dans notre travail, la fonction de hachage et le cryptosystème RSA sont utilisés pour se rassurer de l'authenticité de l'auteur et l'intégrité des données. L'algorithme du cryptosystème RSA est défini comme suit :

1. Deux grands nombres premiers p et q sont générés aléatoirement
2. Calculer n = pq et φ = (p-1) (q-1).
3. Choisir un nombre entier e, 1 <e <φ, tel que PGCD (e, φ) = 1

La clé publique est le couple (n,e) et la clé privée est le couple (n,d) .La clé privée reste toujours gardée par son créateur et la clé publique est donnée à tout le monde qui communique avec le propriétaire de la clé privée correspondante.

Pour le cas de chiffrement et de déchiffrement du message m, la clé publique est utilisée pour chiffrer le message m comme suit :

$C=m^e \bmod n$.La clé privée est utilisée pour déchiffrer le message c qui est chiffré comme suit : $m=c^d \bmod n$

Pour le cas de la signature numérique, la clé privée est utilisée par l'expéditeur pour crypter le message comme suit :

$c=m^d \bmod n$.

La clé publique est utilisée par le récepteur du message crypté pour décrypter le message comme suit : $m=c^e \bmod n$.

Si le récepteur ne parvient pas à décrypter le message alors qu'il est censé avoir utilisé la clé publique de l'expéditeur, il remarque que l'expéditeur n'a pas utilisé la clé privée qui correspond à la clé publique qui est utilisé pour décrypter. Le message a été envoyé par une autre personne. Mais si le message est décrypté .c'est bel et bien l'expéditeur qu'on prétend avoir envoyé le message qu'il l'a envoyé.

De ce fait, le récepteur est rassuré de deux propriétés, l'authenticité de l'auteur ou la non-répudiation.
Mais cette authenticité doit être complémentaire avec l'intégrité des données pour que le système soit plus meilleur. Pour bien assurer la sécurité, la fonction de hachage est utilisée pour que l'intégrité des données soit garantie.
L'expéditeur utilise la fonction de hachage pour produire une empreinte digitale du message. L'empreinte digitale est ensuite cryptée avec la clé privée de l'expéditeur. L'empreinte digitale cryptée et le message sont envoyés. A la réception, le récepteur décrypte le message crypté en utilisant la clé publique. S'il parvient à le décrypter, c'est l'auteur prétendu qu'il l'a envoyé, l'authenticité et la non-répudiation sont vérifiées. Sinon c'est la personne inconnue qui l'a envoyé. La même fonction de hachage est appliquée par le récepteur au message reçu. Les deux empreintes digitales sont comparées, si les deux empreintes digitales sont les mêmes, le message n'a pas été modifié. Mais si les deux empreintes digitales ne sont pas les mêmes, le récepteur conclut que le message a été modifié. La signature numérique permet de s'assurer de l'intégrité du message et l'authenticité de l'auteur et la non-répudiation.
Dans notre projet les valeurs utilisées dans l'algorithme de diffie-hellman sont les suivantes :
Le nombre premier q :
311792034005135522787128926744330811750072666870677383017611284
157797066122504823398389667277316312818720928055219945736732371
391505979080784668006816928233396203275637052744047736179330727
950902939002499508016084700814748359682449138106287573182504097
343798269985666726776340285207039557317480488731965424497730756
102969101197931311758286668651942574806826331377053780726441977
991467758196400433232311373767312604849269793192240857040337647
305053865562850518166153443893161575886869279241665929876063362
174183203393141762492058663067573135232579191271035821589729535
958067430284630882191721241571693671069589663196 79

La valeur de j trouvée est:572
La valeur de h choisie est :2
La valeur de g trouvée par l'interlocuteur A est:
154581500920690333787814075637279493771957093188257399946289410
058060765691582416981447750663276903118498544641260087087094603
146441719506229716346938683229947879232592936 96

Les valeurs a et b pour les deux interlocuteurs sont respectivement 5 et 7.
La valeur de P trouvée est:
178345043450937519034237746097757224321041565450027463086073654
538259921822072758983878889682624930932308370847585808961410916

435941420034208830099899282949502628273664394169595305094577176
387916481109429718585200448866036061738360906996796491860392343
680652610431801367716066643138426626785598839554684222812701992
490898325885216710325739974468911152789504661547674762575524811
411119557688341047808882105794902809973782321705961770227073134
258490811101950496391039769906888421407289227726232911889108243
163632792340877088145457555274651833353035297407032489949325294
568014570122808864613664550179008779851805287348563 89

La valeur g^a mod p est:
177721671249046898584171282682025898986025326973408710134309677
698559475224346741459837215277512842924665992723582828759684143
150457899201063397352881838665632869327177134808785438732966228
329223618281770399713152736013417411140003327138249287751709531
207274096026744237753459967772578385495963105811669679582002742
184842803213507775994963891616512697374862312780663304903342708
803869437654072642199568329379321075794142678931131864553493778
243085360571196936918023521766141465388889853457203837571689838
898412406254282928613046075334358650158675467778134214044722555
549977318886229159577901225933113057540221677601536 46

La valeur g^b mod p est:
240691852051220681213482800596271591580570142119012114591411270
829598869645952146635739889699105492399504370855694953900368346
410624038233261683990189599198637875550736770075275182949828334
571835589869029300740039472942548972867564392238016121426849744
282532932326190201759387694969129839894121489046509631950857015
553136410754992224862905581747817271203723154298773197583454854
510329089379073468625203801972238672802991672411355760946324666
536685364028426521960017791765003360355244263488334836938181296
303714129274333760682045190264597329641865421247419917560693579
610458171931363375833688491914388177457099892956738 6

Comme le nombre premier p est échangé publiquement, la signature est indispensable. Les valeurs utilisées pour la signature avec le cryptosystème RSA sont :
Pour l'interlocuteur A

$p_{A=}$2840412978331462710482172751601421898037180563649649698478547
909489890656279234052478982577763518732724739978213342846412327
624643025178949085818126097829748024945627076029679948742897253
943848480625061002969899876100373505448364170816606266968769 04

167922012687382904483497640142925620138349736161981830190742218
733014259603870023516859127316409892615908058686102825050015266
059109009292844618369215665095419614950819980237204640442217734
440475806024160988737915519756878119938279857038324143901752065
978312517494642437750548248410072407583841156413243749710549394
909684690437185460870485465407134006815241594391582 39

q_A=276514391946501896100966198081039208893126708677461985881378
051948295545148236243762743332604459852727117744882756929649979
742746598159090097125781024446507077804898677554306032013099306
070103327495977808713676267968612247491764898368051774311235009
507162802531623138524523906387905229273610167222716121537000738
327848646445424447353782637684749545446140874422243469904419592
160251810243121856287150697453279270541637581946125757995046537
242543012615009505309810970588122103647573714582716315862975665
061110701882478392951864290086773886843647079046607274703403096
389753753397191391290210142260512237832938016252416 13

e_A = 211

d_A=405735745811612225677390156796705958010857866945755444523316
647264580216658311113820238897000821256907212410242341663633484
643328807392895060571503184392730029955332449146226627102145426
663321839430787909575320982315921522769294606024390421342498729
983299020124397863038935957018141203228715207919384749802077158
469886679691925804195602918256150347276908282706227059880355884
079948293825311964637405500421596205555479784939515781269439949
952297959451114587802232058365564800929091954049306490525421497
713483759759354774443649825937854596889780620260370720845244938
070579926072855959562674579766398179070059016041865141258484200
540606531839632916501613030898504148767587831741367235507838362
905227390036712873425022304938443577952009704349481159352950821
559992952771592836453975001456886741984621614830800005787673544
533668130062599317094061203984705680881389816339400161479733326
695567651257220854446134563233209086347454175851637228284044877
117159035459176810429683206100262740671557874484312030111519217
301286682631291443340858453573789036428741493535423597482215101
984321393906939315890453830821579814091860607157522607939508932
380716935737165532642260929280725917930033580139472659376499934
014897128305340018412595972500572851955

n_A=785415067580276877228709386092706028810009265372058704535961
583236939685457831605652022085019938396398365307900312761712525

318737416145879429179698824833633360739221529998658883656446651
614320257980699531379749791455591204626799650194003476176763596
573175167396770175240509054411264164048246870376056717506773215
019688893715562795277726750018786452068143556431320271878487078
356597155937071784756812482467493572222075546993007613283044306
788393297653075027763953801056276816477416534902785958723522348
784817186323154655115689112595296513245355145641635065122446623
237544627535528508878204920465229502603508737475537164363276925
800302753617086272575390782708186370205640825193270406170239926
909570215722383040766307520528087985527231957724374600204495435
648813998214854952699853114096342356382093883208640038002868856
616625611976094157683169843991846793863708620802499931162473569
712234512386520975636632873441779912174189573421352463978067948
453697674102062565491174613032351552810217567487563572352876118
212341692005085095071311940288965917319334140518676366732757371
349329672780816111516723340433798257930220400389959632920436654
581522291167078376258039593577978725315625665060402706490635603
530567391326149736047012984911714599507

La clé publique de l'interlocuteur A est
(e_A,n_A)

La clé privée de l'interlocuteur A est
(d_A,n_A)

p_B=210259947738463577412310425008806616217765162976348932528919
368255245227672460312727037400780548224947752339320587739394953
190264978755712304803546300377331392847268525235317864056700866
730078155101878273738711999969022193151886531384588726169057194
948375329058182079618341409204896070381199686805021288128739297
054643928765080027775556420985360772065555254060565316926471915
073970381808625110692856701539421713316316803545480925956896060
312928486061727479490714171904828070748951232866646985697989356
018984161786867607715665790648055339178049118833089333937284596
028590796063440355041196349720406506702746216124756 63

q_B=311198987573088421188551101845652151740964985864114419202056
273310707505493576250832145864519695260837830672428833791780764
433211384058140664035854165128693484982497234304051962670565823
949168865629428245375404492300212869350286314366050470573658129
948509403575944345977454775316217855097624935480664023807903687
026846530713111538125904069845770465483799483556179020580060911

102613941311762735637589609716000458756569131900751143207908080
620448595932451533948871409007950856610198834791032262117581781
204796780791556844846403577256162289902033824409261606413610060
339935446635107133650010335859420424242857103514706 77

e_B = 751

d_B=48704220666617329391602287101602346733669557913085903869364 6048556992301116314397746422358560258212189506203157970189973561 8921641604524963182009998512364248369921048737828593994439986913 7527550897463249942651743951200152492723092180798578504712994813 2132152835250333799227925743648120082421650005657458105388773588 2831071607194989624517761428651812520343547129330154602446168147 1641927362366648578196893909193104538901381135921929095665148836 2976319972551427516617296791797576189807601564382878470040774 6727744971722338078503010117499497077457760983224671569551888921 2867691859656085833969315028629192192785564212620935312728390757 8171506528654174438003466537728221557673603080390315208223090790 1918758914271504851514378188862260723671014335692271786679519483 4780587708752827982014245362755762577695486263999945908754558640 2298807021529160095979195191256582142426267306213365928666550094 0552225687433863816937333255544521989999089704895107940487566858 8011800052628678861105280922512378706526896143084954572082623539 6736325236047908586111056940848801594492798288378628767738407199 1624849112031681661908237540431316218671762239447712506683141297 6038547951022722185074007585391964641463169460775120713557899467 7809109554787619996650849959

n_B=65432682863380347715730442957608877275466615371605570314656204376798071223318803704394130819097301852293230513709411926680676389138550952546506073504164321118529708543865997747581832382329665815248649196255691341158052441005429413645402025862324613783049513286762197356023655363101642557899669811941151546863857746084700840848608372322698263282847992450874671322828379911847558502839801676839145305666056894492859955164996560588105267489760392704168047606809477565275606436443389178671708683672808697992710849262316205670049384583937787212357397109154353286824124863798992104833131415583104805904335128411375176526521171403344992301166939387949224899057649883580424278038940055722765319020601945975268838399459677254411581238949930744005770379141075002315147271376949052824357085668394537657017079266281949020095933398580689652935632628311183967202315606953273548137213048956487

023812952556107077909851374400918177464291068469493726188356281
797504450107434446292589008641310382963203401076875323133580082
327571833448621521236848340774946702951907531333756154474478853
628613737754918054840104954021629165574856419407549469655122259
711631719369794885290959249099164659005114972215884442338797862
904181866648941956352567731382920633851

La clé publique de l'interlocuteur B est (e_B,n_B)

La clé privée de l'interlocuteur B est (d_B,n_B)

L'interlocuteur A signe les valeurs à envoyer à l'interlocuteur B et l'interlocuteur B vérifie la signature et vice versa.

En effet, A envoie le nombre premier P signé en utilisant la fonction de hachage et sa clé privée (d_A,n_A)

P=178345043450937519034237746097757224321041565450027463086073654538259921822072758983878889682624930932308370847585808961410916435941420034208830099899282949502628273664394169595305094577176387916481109429718585200448866036061738360906996796491860392343680652610431801367716066643138426626785598839554684222812701992490898325885216710325739974468911152789504661547674762575524811411119557688341047808882105794902809973782321705961770227073134258490811101950496391039769906888421407289227726232911889108243163632792340877088145457555274651833353035297407032489949325294568014570122808864613664550179008779851805287348563 89

L'empreinte digitale de p est est:-1113020474

L'interlocuteur A crypte avec sa clé privée et envoie p l'empreinte digitale de p cryptée.

L'empreinte digitale cryptée est :

270105131226739971035168472884547648300633394213261594963833102447646919066015546779226549511611650625256440279292682079896124717558103821518020330001514596830273350472775318917381513743861685402607920038077138026242500032591196118899272823329743431550652356339762590236519483472882973028681147228629920181810838370837780855268982166526775303739322242290230798203796727887775962920303825243914703398030368799256917283238630606017483063376715331890508021506474192065359098145980870160340009647278613878133445535478463213831892645703569355200963181146070705784426157993066212218755267358460677523357465316593999638507686807059870451397967587999798208229795944778116429286057309451978865600732863028761535980320658990179760656395185852631523867632468837546982823110703036169100652550657450812178166809012749586748421364843946831486141187843409580710807871166678916462915683058869436388642112980629936778526723385850276691956582287292380534146499897296268883280949386711670417809702751952143992591488737069161355

98823963213470654358939760542257075948317119686423272136492486044877329536860062441240259779348170397319459084038676418486166813746156799777814702352387042218093078681273025541117215726475538141379338458115654268371805865905 9

L'interlocuteur B décrypte la valeur reçue et trouve l'empreinte digitale.
Si l'empreinte digitale a une valeur négative B utilise (e_A,-n_A)
Si l'empreinte digitale a une valeur positive B utilise (e_A,n_A)
L'empreinte digitale décryptée est= -1113020474.
B compare les deux empreintes digitales, comme elles sont égales, il est rassuré de l'intégrité, de l'authenticité et de la non-répudiation.
Les deux interlocuteurs procèdent ainsi pour toutes les autres valeurs échangées publiquement. Ces valeurs qui sont échangées publiquement sont g, g^a mod p et g^b mod p .
.La clé secrète trouvée par l'interlocuteur A est
ZA=13338420135970351059737026233022207399185508304407285824597952571326045862528685285385120238691816757212659734447383761948722103314883039040730002575521938648519105471988033801936697014025259409951823816920759574901740571548820355340429569298817155501955244011539116095826730020372719695561330375518565542298952037223459860686499989746749572930375863962213746056322188869377794281347222816788951516635922104206084949308680495641523326061224236013716886414498738957687260833738145544386993345837683741621907947780037262324613804233244340089742324918083461801582001739347776536398693934265356838608410365118070096454363113

La clé secrète trouvée par l'interlocuteur B est ZB=
13338420135970351059737026233022207399185508304407285824597952571326045862528685285385120238691816757212659734447383761948722103314883039040730002575521938648519105471988033801936697014025259409951823816920759574901740571548820355340429569298817155501955244011539116095826730020372719695561330375518565542298952037223459860686499989746749572930375863962213746056322188869377794281347222816788951516635922104206084949308680495641523326061224236013716886414498738957687260833738145544386993345837683741621907947780037262324613804233244340089742324918083461801582001739347776536398693934265356838608410365118070096454363113

III.6. Modèle mathématique

On donne q d'au moins 160 bits de longueur; p d'au moins 1024 bits. On a besoin des paramètres publics (p, q, g). Les parties Alice et Bob doivent

sélectionner les clefs privées dans l'intervalle [2, q − 2] L'algorithme de génération des paramètres publics :

ENTREE: nombres p et q premiers avec des bits de longueur exigée.

SORTIE: paramètres (p, q, g).

1. Génère au hasard un nombre premier q avec la longueur des bits exigée.

2. Choisir un nombre j de longueur des bits égale à bitlen(p) − bitlen(q)

3. Calculer p = jq + 1. Si p pas premier, alors aller à 2.

4. Choisir au hasard un nombre h dans l'intervalle 1 < h < p − 1.

5. Calculer g = h mod p. Si g = 1 alors aller à 4.

6. Retourner (p,q,g)

Algorithme de vérification des paramètres publics :

ENTREE: paramètres (p,q,g).

SORTIE: Accepter ou rejeter les paramètres.

1. Tester 1 < g < p − 1. Si non, retourner "rejeter paramètres" et stop

2. Tester la primalité de q . Si q pas premier alors retourner "rejeter paramètres" et stop

3. Tester la primalité de p. si p pas premier alors retourner "rejeter paramètres" et stop

4. Calculer (p−1) mod q .Si ≠0 alors retourner "rejeter paramètres" et stop.

5. Calculer g^q mod p. Si ≠ 1 alors retourner "rejeter paramètres" et stop

6. Retourner "Accepter paramètres"

Algorithme de génération de la clef à partager par la partie Alice

ENTREE: Paramètres (p,q,g)

SORTIE: La clef privée/publique (a,A)

1. Alice choisit un nombre a dans l'intervalle [2,q−2] .

2. Calcule $A = g^a mod p$.

3. Retourne (a,A). garde a secrète.

Algorithme de génération de la clef à partager par la partie Bob

ENTREE: Paramètres (p,q,g)

SORTIE: La clef privée/publique (b,B)

1. Bob choisit un nombre b dans l'intervalle $[2, q-2]$.

2. Calcule $B = g^b mod p$.

3. Retourne (b,B). garde b secrète.

Algorithme de calcul de la clef secrète partagée par la partie Alice

ENTREE: Paramètres (pq,g),B,a

SORTIE: clef secrète partagée Z_A.

1. Test $1 < B < p$ et $B^q mod\ p = 1$. Si non "échec" et stop.

2. Calcule $Z_A = B^a mod\ p$

3. Retourner Z_A.

Algorithme de calcul de la clef secrète partagée par la partie Bob

ENTREE: Paramètres (pq,g),A,b

SORTIE: clef secrète partagée Z_B.

1. Test $1 < A < p$ et $A^q mod\ p = 1$. Si non "échec" et stop.

2. Calcule $Z_B = A^b mod\ p$

3. Retourner Z_B.

III. 7.Conclusion

Le cryptosystème de Diffie-Hellman est le mieux adapté à nos jours pour sécuriser les communications entre deux parties. Puisque la clé trouvée par échange sécurisé est employée dans les cryptosystèmes symétriques qui sont considérés comme très rapides par rapport aux cryptosystèmes asymétriques. Avec les grands nombres premiers utilisés et la signature numérique, il est impossible qu'une personne qui n'est pas concernée par la communication puisse connaître la clé.

CHAPITRE IV ETUDE DE LA COMPILATION

IV.1. Introduction

Un compilateur est un programme informatique qui transforme un code source écrit dans un langage de programmation (le langage source) en un autre langage informatique (le langage cible).Pour qu'il puisse être exploité par la machine, le compilateur traduit le code source, écrit dans un langage de haut niveau d'abstraction, facilement compréhensible par l'humain, vers un langage de plus bas niveau, un langage d'assemblage ou langage machine. Dans le cas de langage semi-compilé (ou semi-interprété), le code source est traduit en un langage intermédiaire, sous forme binaire (code objet ou bytecode), avant d'être lui-même interprété ou compilé. Inversement, un programme qui traduit un langage de bas niveau vers un langage de plus haut niveau est un décompilateur[2 .4].La compilation est réalisée en 6 phases qui sont les trois phases d'analyse et les trois phases de synthèse. L'analyse décompose et identifie les éléments et relations du programme source et construit son image (représentation hiérarchique du programme avec ses relations), la synthèse qui construit à partir de l'image un programme en langage cible. [6]

L'étape d'analyse comprend les trois phases suivantes :

- L'analyse lexicale

- L'analyse syntaxique

- L'analyse sémantique ou contextuelle

L'étape de synthèse comprend les trois phases suivantes :

- Génération du code intermédiaire

-Optimisation du code intermédiaire

-Production du code final [7]

IV.2. Analyse lexicale

Au départ, le programme en langage source se présente comme une suite de caractères. Le premier travail est effectué par l'analyseur lexical, qui va découper cette séquence de caractères en mots constitutifs du langage. Le résultat est une séquence de mots, avec leurs types et leurs valeurs autant qu'il lui est possible de les reconnaître [8].
Un ensemble fini de symboles est appelé alphabet (par la suite, noté Σ). [9]

Une chaîne ou un mot sur un alphabet Σ est une séquence finie de symboles extraits de cet ensemble [10].Lexème est une chaîne de caractères tandis qu'une unité lexicale est un type d'un lexème [11].
Les concepts de théorie de langage comme les langages réguliers, les automates finis déterministes, les expressions régulières, etc. sont utilisés dans cette étape.

IV.2.1. Automate fini déterministe

Pour notre travail un automate fini déterministe est utilisé pour reconnaître les mots du langage. Un automate fini déterministe est un quintuplet $M = \{Q, \Sigma, \delta, q0, F\}$ où

• Q est un ensemble fini d'états

• Σ est un ensemble fini de symboles (un alphabet)
• $\delta : Q \times \Sigma \rightarrow Q$ est une fonction de transitions
• $q0 \in Q$ est l'´etat initial
• $F \subseteq Q$ est l'ensemble (fini) des états finaux [12]

Soient les instructions suivantes issues de notre programme :

Int ix=(x) ?1 :0 ; Int ix=(y) ?1 :0 ;Int ix=(z) ?1 :0 ;

Dans le tableau 7 montre les unités lexicales et les lexèmes.

Tableau 7 Lexèmes et unités lexicales correspondantes

Lexèmes	Unités lexicales
Int	Mot clés du langage
X, y, z, ix, iy, iz	identifiants
(	Parenthèse ouvrante
)	Parenthèse fermante
=	Opérateur d'affectation
?	Opérateur
0	Nombre
;	Point virgule

1	Nombre
:	Opérateur

L'automate suivant est utilisé pour reconnaître ces mots.

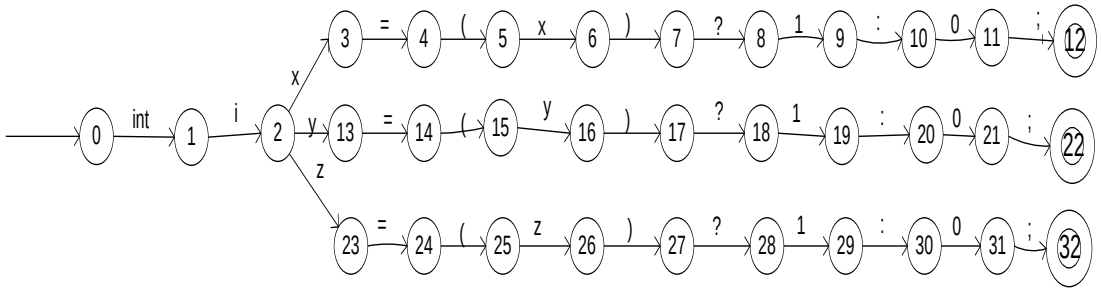


Figure 3 Automate fini déterministe

IV.2.2.Minimisation de l'automate

L'automate de la figure 3 étant déterministe on peut le minimiser pour avoir un autre automate ayant le nombre d'états inferieur au nombre d'états de l'automate fini déterministe de la figure 3.

Ici la méthode utilisée est la méthode des relations des états équivalents. Le tableau 8 appelé table de transition, les lignes représentent les états tandis que les colonnes représentent les lexèmes.

Tableau 8 Table de transition

	int	i	x	y	z	=	(	)	?	1	:	0	;
0	1												
1		2											
2			3	13	23								
3						4							
4							5						
5			6										

6								7					
7									8				
8										9			
9											10		
10												11	
11													12
12													
13						14							
14							15						
15				16									
16								17					
17									18				
18										19			
19											20		
20												21	
21													22
22													
23						24							
24							25						
25					26								
26								27					
27									28				
28										29			
29											30		

30												31	
31													32
32													

$\equiv_0${0,1,2,3,4,5,6,7,8,9,10.11,13,15,16,17,18,19,20,21,23,25,26,27,28,29,30,31} {12,22,32}

$\equiv_1${0},{1},{2},{3,13,23},{4 ,14,24},{5},{6,16,26},{7,17,27},{8,18,28},{9,19, 29},{10,20,30},{11,21,31},{12,22,32},{15} ;{25}

$\equiv_2${0},{1},{2},{3,13,23},{4},{14},{24},{5},{6,16,26},{7,17,27},{8,18,28},{9 ,19,29},{10,20,30},{11,21,31},{12,22,32},{15} ;{25}

$\equiv_3${0},{1},{2},{3},{13},{23},{4},{14},{24},{5},{6,16,26},{7,17,27},{8,18,2 8},{9,19,29},{10,20,30},{11,21,31},{12,22,32},{15} ;{25}

$\equiv_4${0},{1},{2},{3},{13},{23},{4},{14},{24},{5},{6,16,26},{7,17,27},{8,18,2 8},{9,19,29},{10,20,30},{11,21,31},{12,22,32},{15} ;{25}

L'automate trouvé par la minimisation est la suivante

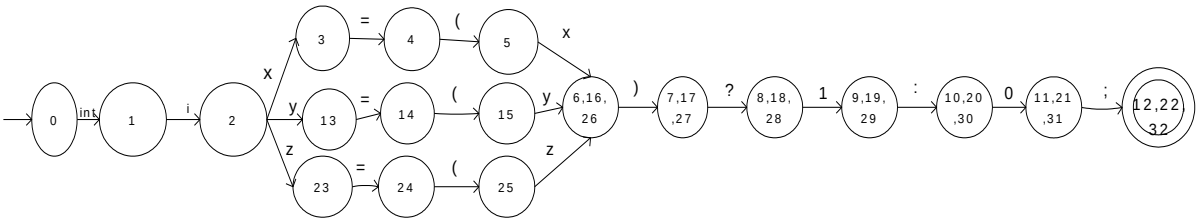


Figure 4 Automate minimisé

IV.2.3. Expression régulière

Les expressions régulières sont une façon déclarative de décrire la formation des chaînes d'un langage [13].Les noms des états de l'automate de la figure peuvent être changés pour mieux déterminer l'expression régulière.

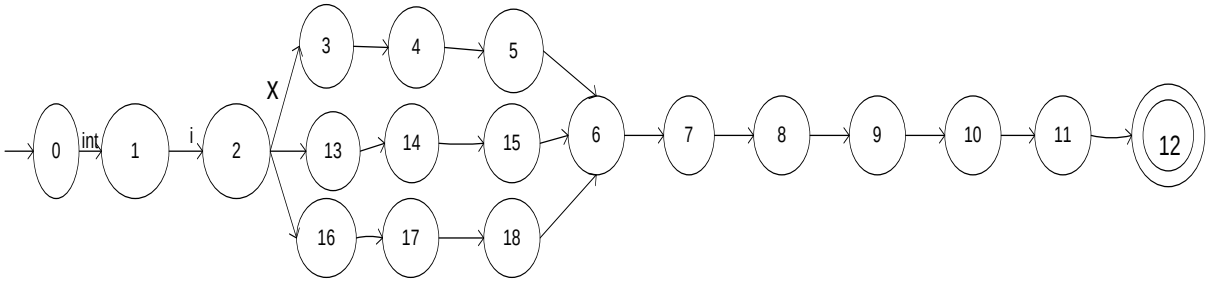


Figure 5 Automate minimisé et renommé

Le système d'équations suivant est résolu pour avoir l'expression régulière.

1) $L_0 = intL_1$
2) $L_1 = iL_2$
3) $L_2 = xL_3 + yL_{13} = zL_{16}$
4) $L_3 = =L_4$
5) $L_4 = (L_5$
6) $L_5 = xL_6$
7) $L_{13} = =L_{14}$
8) $L_{14} = (L_{15}$
9) $L_{15} = yL_6$
10) $L_{16} = =L_{17}$
11) $L_{17} = (L_{18}$
12) $L_{16} = zL_6$
13) $L_6 =)L_7$
14) $L_7 = ?L_8$
15) $L_8 = 1L_9$
16) $L_9 = :L_{10}$
17) $L_{10} = 0L_{11}$

18) $L_{11} = ;L_{12}$

19) $L_{12}=\mathcal{E}$

Ce système d'équations est résolu par substitution

$L_{12}=\mathcal{E}$

$L_{11} = ; \mathcal{E}$

$L_{11} = ;$

$L_{10} = 0;$

$L_{9} = :0;$

$L_{8} = 1:0;$

$L_{7} = ?1:0;$

$L_{6} =)?1:0;$

$L_{18} = ≈)?1:0;$

$L_{17} = (≈)?1:0;$

$L_{16} = =(≈)?1:0;$

$L_{15} = y)?1:0;$

$L_{14} = (y)?1:0;$

$L_{13} = =(y)?1:0;$

$L_{5} = x)?1:0;$

$L_{4} = (x)?1:0;$

$L_{3} = =(x)?1:0;$

$L_{2}=x[^=(x)?1:0;+ =(y)?1:0;+ =(≈)?1:0;]$

$L_{1}=ix[^=(x)?1:0;+ =(y)?1:0;+ =(≈)?1:0;]$

$L_{0}=int\ ix[^=(x)?1:0;+ =(y)?1:0;+ =(≈)?1:0;]$

L'expression régulière est alors

$int\ ix[^=(x)?1:0;+ =(y)?1:0;+ =(≈)?1:0;]$

ou

int ix[=(x)?1:0;/ =(y)?1:0;/ =(≈)?1:0;]

IV.3.Analyse syntaxique

Le procédé qui consiste à trouver une dérivation d'un mot ω d'un langage algébrique s'appelle l'analyse syntaxique [14].L'analyse syntaxique c'est la phase qui suit immédiatement la phase d'analyse lexicale et est la deuxième phase de la compilation : Cette phase est située entre l'analyse lexicale et l'analyse sémantique.

L'analyse syntaxique a pour but de vérifier que le texte d'entrée est conforme à la grammaire, d'indiquer les erreurs de syntaxe et éventuellement de poursuivre l'analyse après une erreur (reprise sur erreur) et de construire une représentation intermédiaire pour les autres modules du compilateur [15].
Le résultat est confié à l'analyseur sémantique et ensuite le générateur du code pour terminer l'analyse et générer le code correspondant.

Les grammaires hors-contexte et les automates à pile sont deux outils théoriques utilisés dans la phase d'analyse syntaxique. Une chaine est acceptée si elle appartient au langage engendré par la grammaire. La grammaire hors contexte est aussi appelée grammaire algébrique, ou grammaire non contextuelle et elle est donnée par un quadruplet :

$G = (V_N, V_T, P, S)$ où

- V_N est l'ensemble des symboles non-terminaux;
- V_T est l'ensemble des symboles terminaux;
- P est l'ensemble des productions de la grammaire, avec $P \subset V_N \times (V_N + V_T)*$.
- $S \in V_N$ est le symbole de départ ou l'axiome de la grammaire.

Les 3 grandes méthodes d'analyse syntaxique sont: La méthode universelle, la méthode ascendante et la méthode descendante.

IV.3.1. Méthode universelle

Dans la méthode universelle toutes les dérivations sont essayées à partir de l'axiome, jusqu'à trouver le mot.

IV.3.2.Analyse syntaxique descendante.

On part de l'axiome S et on essaye de trouver une dérivation qui aboutit à un mot. L'une des méthodes utilisées est la méthode des grammaires dites **L**eft to right **L**eftmost derivation(LL).LL(1) est utilisé dans notre travail. Le chiffre 1 signifie que c'est un seul symbole de prévision à chaque étape. [16] La grammaire LL(1) va permettre de déterminer laquelle des règles qu'il faut choisir si un symbole non terminal a plusieurs règles, mais également si un non terminal est annulable, elle permet de déterminer s'il faut considérer la production vide ou non. Un langage hors-contexte est dit LL(1) si et seulement s'il existe une grammaire LL (1) qui le reconnaît. L'ensemble des langages LL(1) est strictement inclus dans l'ensemble des langages hors-contexte. Pour ces grammaires on doit déterminer deux ensembles : l'ensemble premier et Suivant. Premier est pris comme l'ensemble des symboles terminaux qui peuvent débuter une dérivation. Pour $\alpha \in (V_T \cup V_N)$, Premier(α) contient l'ensemble de V_T susceptibles de commencer un mot de V_T dérivé de α. Suivant est l'ensemble des symboles terminaux qui peuvent suivre une occurrence dans une dérivation. Pour $\alpha \in (V_T \cup V_N)$, Suivant(α) contient l'ensemble des terminaux de V_T susceptibles de suivre α dans un mot de V_T dérivé de l'axiome S.

Soit la grammaire :

$E \rightarrow E+T/T$

$T \rightarrow T*F/F$

$F \rightarrow (E)/id$

Comme la présence des règles de grammaire récursive gauche cause des phénomènes de bouclage infini, la suppression de la récursivité à gauche est indispensable. Une variable est récursive gauche s'il existe une règle de la grammaire dont le membre droit débute par cette variable (ce cas est dit récursivité directe) ou pour laquelle un mot dérivé débute par cette variable (cas de la récursivité indirecte). La suppression de la récursivité gauche est procédée de la façon suivante pour cette grammaire.

Considérons par exemple une grammaire ayant deux règles de la forme

$A \rightarrow A\alpha$

$A \rightarrow \beta$

$avec\ \lambda(\alpha) \neq \{\mathcal{E}\}.$

Transformation de la grammaire

On introduit un nouveau non terminal A' et on remplace les règles

$A \rightarrow A\alpha$

$A \rightarrow \beta$

Par

$A \rightarrow \beta\ A'$

$A' \rightarrow\ \mathcal{E}/\alpha\ A'$

La grammaire devient

$E \rightarrow TE'$

$E' \rightarrow\ +\ TE'/\mathcal{E}$

$T \rightarrow\ FT'$

$T' \rightarrow\ *\ FT'/\mathcal{E}$

$F \rightarrow\ (E)/id$

Déterminons les ensembles Premier et Suivant :

$Premier(E) = \{(, id\}$

$Premier(T) = \{(, id\}$

$Premier(T') = \{*, \mathcal{E}\}$

$Premier(F) = \{(, id\}$

$Premier(E') = \{+, \mathcal{E}\}$

$Suivant(E) = \{\$,)\}$

$Suivant(E') = \{\$,)\}$

$Suivant(T) = \{+, \$,)\}$

$Suivant(T') = \{+, \$,)\}$

$Suivant(F) = \{*,+,\$,)\}$

Le tableau 9 est une table de prédiction.les colonnes représentent les symboles terminaux tandis que les lignes représentent les symboles non terminaux.

Tableau 9 Table de prédiction

	id	+	*	(	)	$
E	E⟶TE'			E⟶ TE'		
E'		E'⟶+ TE'			E' ⟶ Ɛ	E'⟶ Ɛ
T	T⟶ FT'			T⟶ FT'		
T'		T'⟶ Ɛ	T'⟶ * FT'		T'⟶ Ɛ	T' ⟶ Ɛ
F	F⟶ id			F ⟶(E)		

Soit W=id+id*id le mot à analyser.

Le tableau 10 montre les étapes pour aboutir à un arbre syntaxique ayant une table de prédiction. La colonne pile montre les états de la pile à chaque étape, la colonne entrées contient le mot à analyser pour chaque étape, la colonne action montre les productions qu'il faut prendre.

Tableau 10 Table de réduction

Pile	Entrées	Action
E$	Id+id*id$	E ⟶ TE'
T E'$	Id+id*id$	T⟶ FT'
FT'E$	Id+id*id$	F ⟶ id
idT' E'$	Id+id*id$	
T' E'$	+id*id$	T' ⟶ Ɛ
Ɛ E'$	+id*id$	
E'$	+id*id$	E'⟶ + TE'
+ TE'$	+id*id$	
TE'$	id*id$	T⟶ FT'
FT' E'$	id*id$	F ⟶ id
idT' E'$	id*id$	
T' E'$	*id$	T'⟶ * FT'
* FT'E'$	*id$	
FT'E'$	id$	F⟶ id
idT'E'$	id$	
T'E'$	$	T' ⟶Ɛ
Ɛ E'$	$	
E'$	$	E'⟶ Ɛ
Ɛ '$	$	
$	$	

Puisque dans la pile reste $ et du coté du mot le caractère restant est $, on conclut que le mot est reconnu. On est parti de la racine jusqu'à représenter le mot entier sur les feuilles. Noter que le mot trouvé contient des Ɛ qui sont des mots vides, les Ɛ sont supprimés. L'algorithme nous permet également de représenter progressivement l'arbre syntaxique suivant.

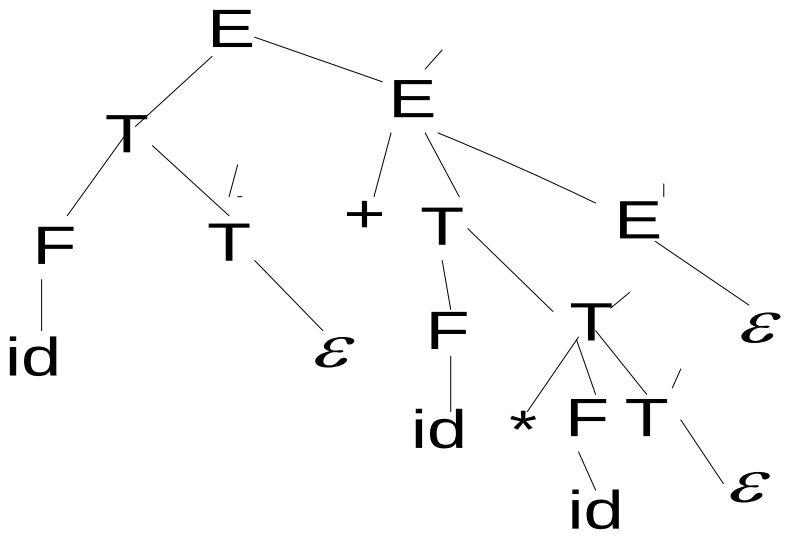


Figure 6 Arbre syntaxique

Le mot trouvé est idƐ+id*idƐƐ.

Après avoir supprimé les mots vides symbolisés par Ɛ, le mot devient id+id*id.

IV.3.3.Analyse syntaxique ascendante

L'analyse syntaxique ascendante ou par décalage réduction a pour but de construire un arbre d'analyse à partir des feuilles pour une chaîne d'entrée donnée w. La racine est construite après avoir construit tous les nœuds inférieurs. A chaque étape de réduction une sous-chaîne correspondant à la partie droite d'une règle de production est remplacée par la variable de la partie gauche.[17] L'analyse syntaxique ascendante part de la chaîne à analyser.Analyseur ascendant prédictif LR(k) est utilisé. Dans notre cas nous allons utiliser K=1 et k=0. (LR(0), SLR(1), LR(1), LALR(1)) sont des algorithmes et des variantes de LR(k) utilisés et basés sur les opérations de décalage/réduction. Pour l'opération de décalage, des terminaux dans le mot d'entrée sont lus. Pour l'opération de réduction, on remplace une partie droite par une partie gauche de règle de production. Pour analyser le mot donné pour une grammaire G donnée, on dispose à chaque instant, d'un suffixe du mot d'entrée et une pile contenant des terminaux et des non-terminaux. Initialement, le suffixe est le mot à analyser et la pile est vide. L'analyse

réussit si toute la chaine a été entièrement lue et si la pile contient l'axiome. Pour une grammaire donnée, on fait d'abord l'augmentation de la grammaire .pour une chaîne donnée le premier terminal du mot d'entrée est effacé et est placé au sommet de la pile, il s'agit d'un décalage. On applique la réduction chaque fois que les n premiers symboles sur la pile forment un membre droit de règle de production, on les dépile et on empile le membre gauche correspondant.
Prenons l'exemple de la chaîne w se trouvant dans notre programme. W=id+id

Soit la grammaire

$$E \longrightarrow E+T/T$$

$$T \longrightarrow id$$

La grammaire augmentée est :

$$E' \longrightarrow E$$

$$E \longrightarrow E+T$$

$$T \longrightarrow id$$

$$E \longrightarrow T$$

La première règle de production est :

$$E \longrightarrow E+T$$

La deuxième règle de production est

$$E \longrightarrow T$$

La troisième règle de production est :

$$T \longrightarrow id$$

La quatrième règle de production est



On commence alors l'analyse en lisant de gauche vers à droite la chaine id+id$.



Figure 7 Premier état de la pile

Au sommet de la pile on lit le premier caractère id et la chaîne restante est +id $.on va ensuite faire la réduction, en remplaçant le caractère id se trouvant dans la pile par T suivant la troisième règle de production. Le contenu de la pile est représenté par la figure 8 :



Figure 8 Deuxième état de la pile

Le non terminal T se trouve dans la pile la chaîne restante est toujours +id $

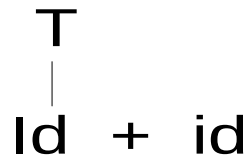


Figure 9 Réduction de id en T

Et puis le non terminal T se trouvant dans la pile va être remplacé par le non terminal E suivant la deuxième règle de reproduction. Le contenu de la pile est représenté par la figure 10



Figure 10 Troisième état de la pile

La pile contient cette fois –ci le non terminal E et la chaine non encore analysée +id$.

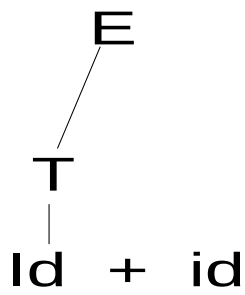


Figure 11 Réduction de T en E

Et puis on va faire le décalage en lisant le caractère + .La pile contient cette fois –ci le non terminal E et le caractère +

Le contenu de la pile est représenté par la figure 12



Figure 12 Quatrième état de la pile

Dans la pile se trouve E+, le mot non encore analysé est id$.

Et graphiquement la situation reste comme avant.

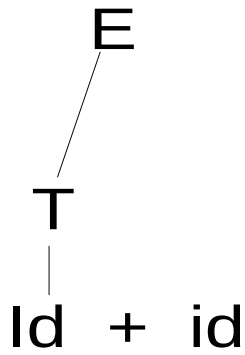


Figure 13 Précédent état de l'arbre syntaxique

On fait encore une réduction pour le caractère id .La pile va contenir le non terminal E et les deux terminaux +id.

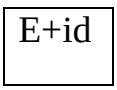


Figure 14 Cinquième état de la pile

Graphiquement on a toujours

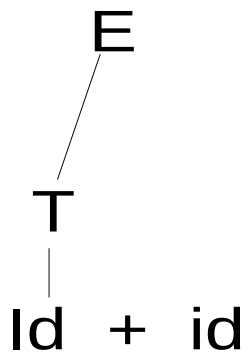


Figure 15 Etat précédent de l'arbre syntaxique

La chaine non encore analysée est $.Comme dans la pile il y a un caractère id on fait la réduction suivant la troisième règle de production, et le contenu de la pile est représenté par la figure 16.

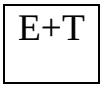


Figure 16 Sixième état de la pile

Dans la pile le mot id est remplacé par le non terminal T.

L'arbre syntaxique devient

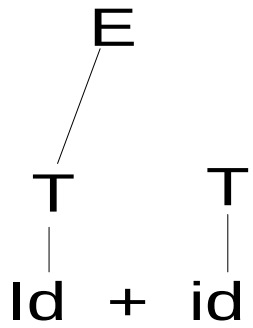


Figure 17 Réduction de id en T

Comme dans la pile il reste E+T, on applique la réduction en utilisant la première règle de production et on a les situations suivantes :

Dans la pile reste le non terminal E, la chaine non encore analysée est $.



Figure 18 Etat 7 de la pile

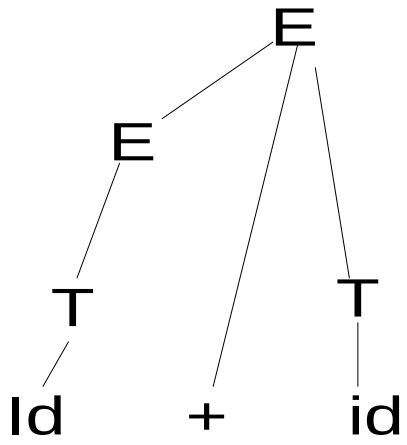


Figure 19 Réduction de T + E en E

Et enfin en analysant le mot $ on applique la réduction suivant la quatrième règle de production.

Comme on a abouti à l'axiome 'on conclut que la chaine id+id est engendrée par la grammaire. L'arbre syntaxique est représenté par la figure 20.

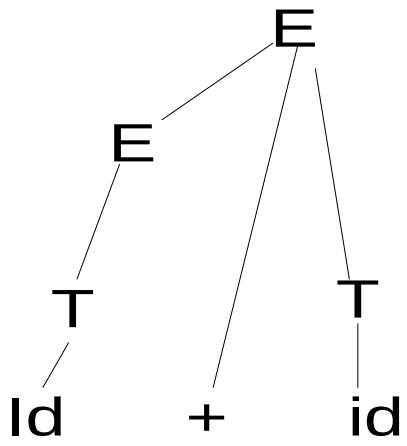


Figure 20 Arbre syntaxique complet

Mais il faut un algorithme permettant de décider à chaque étape si on doit réduire (et par quelle règle) ou décaler. On utilise un automate déterministe appelé automate LR(0) qui précise quoi faire en fonction du contenu de la pile et la première lettre de l'entrée. Les règles pointées sont utilisées pour construire les automates LR(0).

Pour une règle suivante :

$$V \longrightarrow AB$$

Les règles pointées sont :

$$V \longrightarrow A.B \qquad V \longrightarrow .AB \qquad V \longrightarrow AB.$$

La première règle pointée signifie que A est déjà lu mais B n'est pas encore lu. La deuxième règle pointée signifie que A n'est pas encore lu B n'est pas non plus lu. La troisième règle pointée signifie que A et B sont tous déjà lus. Lors de la construction de l'automate, on ne considère bien-sûr que les états accessibles.

Soit id+id le mot à analyser,

Soit la grammaire

$$E \longrightarrow E+T/T$$

$$T \longrightarrow id$$

La grammaire augmentée est la suivante :

$$E' \longrightarrow E$$

$$E \longrightarrow E+T$$

$$T \longrightarrow id$$

$$E \longrightarrow T$$

La construction de l'automate est totalement indépendante du mot d'entrée (pas de symbole de prévision), d'où son nom : LR(0).

On construit l'automate tel que :

Si l'état q contient une règle pointée complète A → B•, alors on réduit en utilisant la règle A → B.Si le mot courant débute par le terminal a de Σ et s'il existe une transition δ(q,a) = q', alors on décale.

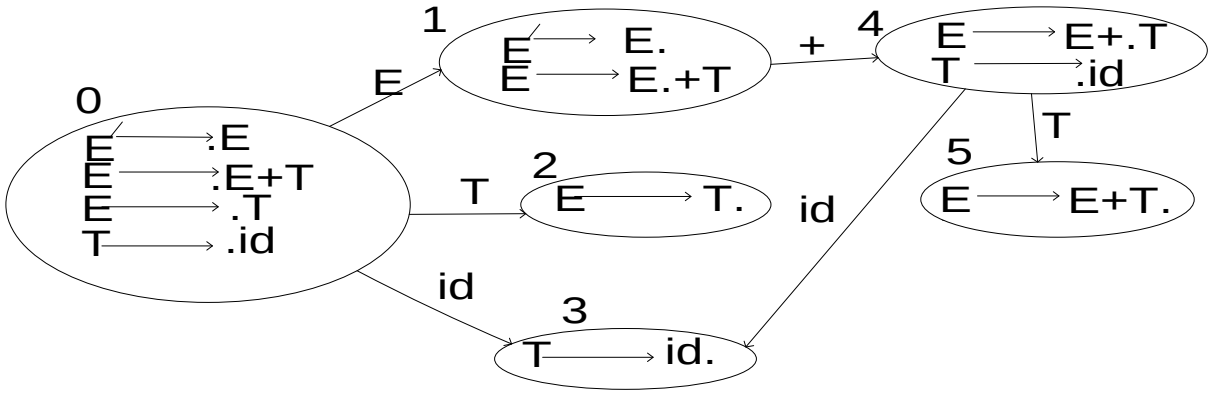


Figure 21 Automate

Si pour tout état q de l'automate, q ne contient au plus qu'une seule règle pointée complète (pas de conflit réduction/réduction) et si q contient une règle pointée complète alors pour aucun a de Σ il n'existe de transition q'= δ(q,a) (pas de conflit décalage/réduction), alors l'automate est dit LR(0)-consistant.On utilise toujours une pile. Initialement, la pile contient l'état initial 0. L'automate LR(0) est en fait codé dans deux tables,une table Action et une table Successeur:

Action : Q × (Σ ∪ {$}) → un ensemble d'actions

Les actions sont : décaler, réduire, accepter ou erreur.

Successeur : Q × V → Q est la restriction de la fonction de transition aux non-terminaux.

La table action est le tableau 11, les colonnes sont des symboles terminaux, et les lignes sont des numéros des états de l'automate on la remplit comme suit: lorsqu'il s'agit d'un décalage, on écrit décalage suivi du numéro de l'état de l'automate vers lequel on décale, s'il s'agit d'une réduction on écrit la production. On met "accepter" dans la case (q,$) où q est l'état contenant la production ajoutée pour l'augmentation de la grammaire, et enfin dans toutes les cases encore vide sont remplies avec le mot erreur.

Dans le tableau 11 qui est une table d'action, les colonnes représentent les symboles terminaux tandis que les lignes représentent les états de l'automate.

Tableau 11 Table d'action pour la méthode LR

	id	+	$
0	Décalage 3	erreur	Erreur
1	erreur	Décalage 4	Accepter
2	E ⟶ T	E ⟶ T	E ⟶ T
3	T ⟶ id	T ⟶ id	T ⟶ id
4	Décalage 3	erreur	Erreur
5	E ⟶ E+T	E ⟶ E+T	E ⟶ E+T

La table successeur où les colonnes sont des non terminaux et les lignes sont des numéros des états de l'automate, elle est remplie comme suit : si un non terminal permettant de quitter un état de départ vers l'autre état, on mentionne la valeur de l'état dans la case de la table correspondant à l'intersection de la colonne du non-terminal concerné et de la ligne du numéro de l'état de départ. Les autres cases de la table restent vides. Dans le tableau 12 qui est une table de successeur, les colonnes représentent les symboles non terminaux tandis que les lignes représentent les états de l'automate.

Tableau 12 Table de successeur pour la méthode LR

	E	T
o	1	2
1		
2		
3		
4		5
5		

L'analyse est faite en utilisant une pile, la pile contient la valeur zéro au départ. Le mot est alors analysé systématiquement comme suit : Le symbole $ est ajouté au mot à analyser. On lit le numéro de la ligne au départ qui est 0 et le caractère de la chaîne à analyser, et on fait un décalage ou une réduction respectivement si la case d'intersection montre l'opération de décalage ou de réduction.

Analysons la chaîne id+id$.au départ la pile contient la valeur 0 qui l'état 0 de l'automate.



Figure 22 Premier état de la pile

Le premier caractère à analyser est id, alors la table action nous montre que 0 et id ont pour opération décalage vers l'état 3, on remplit la table avec cette opération, en écrivant d pour symboliser le décalage. Le contenu de la pile est représenté par la figure 23

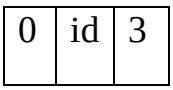


Figure 23 Deuxième état de la pile

La pile contient 0, id et 3 et le mot non encore analysé est +id$.On avance en lisant le caractère +.comme dans la pile la valeur qui est à la tête est 3, on regarde dans la table action 3 comme numéro de la ligne et + comme la colonne l'opération correspondante est la réduction, la réduction est alors faite suivant la production indiquée dans la table. Dans la pile id est remplacé par T comme le stipule la règle de production dans la table action éventuellement la valeur de décalage de id est effacée.

Le contenu de la pile est représenté par la figure 24

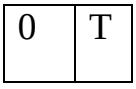


Figure 24 Etat 3 de la pile

La pile contient 0 et T, le mot non encore analysé est +id$.Ayant 0 et T dans la pile, on regarde dans la table Successeur ce qui suit T étant en 0.on a alors 2, la valeur 2 est ajoutée dans la pile. Le contenu de la pile est représenté par la figure 25.

0	T	2

Figure 25 Etat 4 de la pile

Etant dans 2 on veut lire le caractère +.La table Action nous montre qu'il s'agit de faire la réduction .T et 2 sont alors remplacés par E. Le contenu de la pile est représenté par la figure 26.

0	E

Figure 26 Etat 5 de la pile

Le mot non encore analysé est +id$.Graphiquement, l'arbre syntaxique se présente comme montré à la Figure 27.

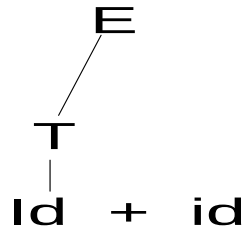


Figure 27 Réduction de T en E

Avec les éléments se trouvant dans la pile 0 et E, dans la table successeur nous donne 1.la pile contient alors 0, E et 1,

0	E	1

Figure 28 Etat 6 de la pile

On veut lire + étant en 1, la table action nous montre qu'il s'agit de faire un décalage. Le mot non encore analysé est id$

Le contenu de la pile est représenté par la figure 29.

0	E	1	+	4

Figure 29 Etat 7 de la pile

La lecture concerne cette fois id étant en 4, la table action nous montre qu'on fait un décalage vers l'état 3 de l'automate. Le mot qui n'est pas encore analysé est $. Le contenu de la pile est représenté par la figure 30.

0	E	1	+	4	id	3

Figure 30 Etat 8 de la pile

En lisant le $ qui reste et étant dans 3, la table action nous montre qu'on fait une réduction en suivant la production montrée dans la table.id et 3 sont remplacés par T. Le contenu de la pile est représenté par la figure 31.

0	E	1	+	4	T

Figure 31 Etat 9 de la pile

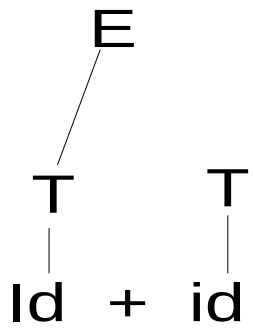


Figure 32 Réduction de id en T

Dans la table successeur 4 et T nous donnent 5. Le contenu de la pile est représenté par la figure

0	E	1	+	4	T	5

Figure 33 Etat 10 de la pile

Comme le caractère $ n'est pas encore lu, on essaie de le lire. La table action nous montre que étant en 5 en essayant de lire $ on fait un décalage en suivant la production montrée dans la table action. Le $ est alors lu. Dans la pile E+T est remplacé par E.

0	E

Figure 34 Etat 11 de la pile

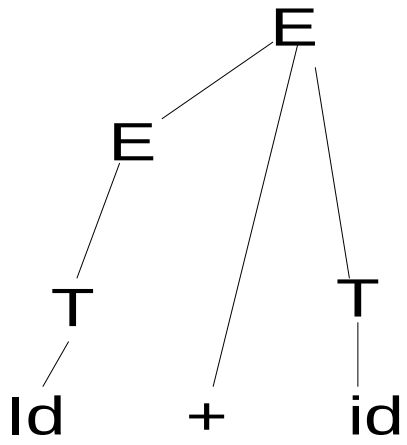


Figure 35 Réduction de E+T en E

En considérant les éléments de la pile et la table successeur, 0 et E donnent 1.

0	E	1

Figure 36 Etat12 de la pile

Etant dans 1 en lisant le caractère $, la table action nous montre qu'on trouve accepter comme action. Le mot est alors accepté, il est engendré par la grammaire.

L'arbre syntaxique est représenté à la figure 37.

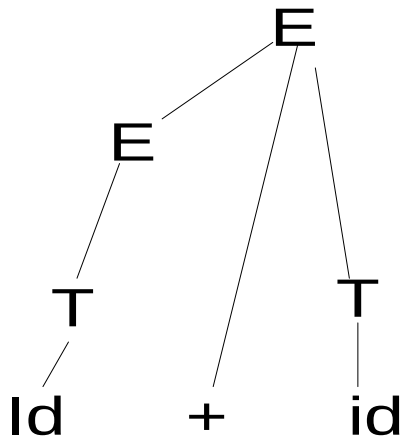


Figure 37 Arbre syntaxique obtenue par LR

LR(0) n'est pas efficace pour tous les mots avec des grammaires respectives, ils peuvent présenter des conflits de réduction –réduction si un état présente deux cas de réduction à la fois ou des conflits de décalage-réduction les deux

situations en même temps. Si de telles situations se présentent on va utiliser les algorithmes d'analyse Comme SLR(0).

Prenons le mot extrait du programme id+id*id, et la grammaire suivante :

E ⟶ E+T/T
T ⟶ T*F/F
F ⟶ id

La grammaire augmentée est la suivante :

E' ⟶ E
E ⟶ E+T/T
T ⟶ T*F/F
F ⟶ id

L'automate correspondant est le suivant :

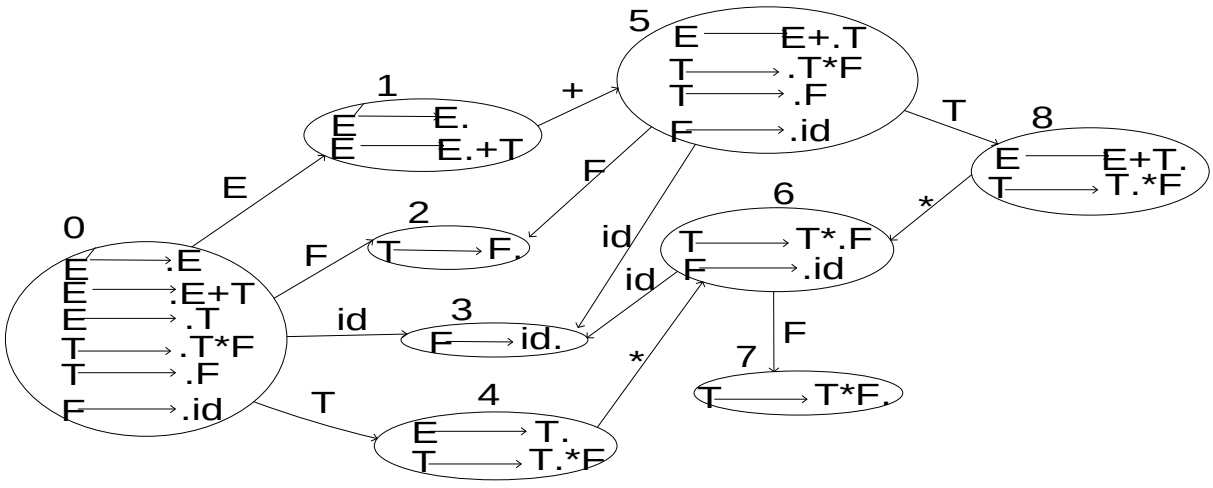


Figure 38 Automate SLR

En essayant de faire l'analyse par LR(0), il y a des conflits de décalage/réduction. Pour l'état 4 ça nous montre qu'il faut faire la réduction vers l'état alors qu'il y a une règle pointée complète qui fait qu'on devrait faire la réduction, c'est une situation non déterministe pour laquelle on n'est pas sûr de quelle opération il faut choisir entre décalage et réduction. C'est le même cas pour l'état 8, on a le conflit de décalage/réduction. Ce problème des conflits décalage/réduction est résolu en regardant si le terminal présent en tête du mot restant à analyser appartient dans l'ensemble des Suivants du non-

terminal en membre gauche de la règle utilisable pour la réduction. Si ce n'est pas le cas, donc la seule opération possible est le décalage. Ce qui est indispensable alors est de déterminer l'ensemble Premier et l'ensemble Suivant au préalable.

$Premier(F)=\{id\}$

$Premier(T)=\{id\}$

$Premier(E)=\{id\}$

$Premier(É)=\{id\}$

$Suivant(É)=\{\$\}$

$Suivant(E)=\{\$,+\}$

$Suivant(T)=\{\$,+,*\}$

$Suivant(F)=\{\$,+,*)$

Dans le tableau 13 qui est une table d'action, les colonnes représentent les symboles terminaux tandis que les lignes représentent les états de l'automate.

Tableau 13 Table d'action pour la méthode SLR

	+	*	id	$
0	Erreur	Erreur	Décalage 3	Erreur
1	Décalage 5	Erreur	Erreur	Accepter
2	T⟶ F	T⟶ F	Erreur	T⟶ F
3	F⟶id	F⟶id	Erreur	F⟶id
4	E⟶ T	Décalage 6		E⟶ T
5	Erreur	Erreur	Décalage 3	Erreur
6	Erreur	Erreur	Décalage 3	Erreur
7	T⟶T*F	T⟶T*F	Erreur	T⟶T*F
8	E⟶E+T	Décalage 6	Erreur	E⟶E+T

Dans le tableau 14 qui est une table de successeur, les colonnes représentent les symboles terminaux tandis que les lignes représentent les états de l'automate.

Tableau 14 Table de successeur pour la méthode SLR

	E	T	F
0	1	4	2
1			
2			
3			
4			
5		8	2
7			
8			
9			

La pile est toujours utilisée avec comme contenu au départ la valeur 0 qui est l'état de départ. Au départ le contenu de la pile est représenté par la figure 39



Figure 39 Etat 1 de la pile

La pile contient 0 comme valeur ; la chaîne à analyser est id+id*id$, Etant en 0 et en lisant id, le décalage vers l'état 3 est fait d'après la table action, le contenu de la pile devient 0, id, 3.Le contenu de la pile est représenté par la figure 40.

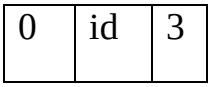


Figure 40 Etat 2 de la pile

On poursuit la lecture de la chaîne restante, + est le caractère qui est concerné par l'analyse, étant dans 3 la table action nous montre qu'on fait une réduction ; id est alors remplacé par le symbole non terminal F. La situation devient :

0	F

Figure 41 Etat 3 de la pile

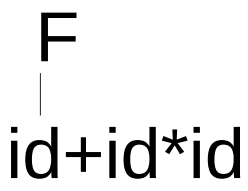


Figure 42 Réduction de id en F

La table successeur indique que dans 0 , F avance vers 2. Le contenu de la pile est représenté par la figure 43.

0	F	2

Figure 43 Etat 4 de la pile

La chaîne qui n'est pas encore analysée est toujours +id*id$.Etant dans 2 en essayant de lire +, la table action montre qu'if faut faire une réduction en remplaçant F par T.

0	T

Figure 44 Etat 5 de la pile

La chaîne qui n'est pas encore analysée est toujours +id*id$

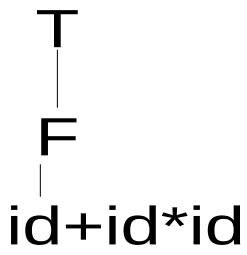


Figure 45 Réduction de F en T

Etant dans 0 et le symbole non terminal est T on a la valeur 4.

0	T	4

Figure 46 Etat 6 de la pile

Le caractère à analyser est +, la réduction est alors faite en utilisant la production montrée dans la table action. Le symbole non terminal T est remplacé par le symbole non terminal E dans la pile.

0	E

Figure 47 Etat 7 de la pile

La chaîne restante est +id*id$, graphiquement, l'arbre syntaxique est représenté comme montré à la figure 48.

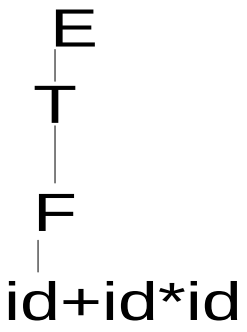


Figure 48 Réduction de T en E

Ayant 0 et E on a comme valeur 1 d'après la table Successeur .1a pile contient alors 0, E et 1.

0	E	1

Figure 49 Etat 8 de la pile

Etant dans 1 comme le montre la pile et en essayant de lire le caractère +, c'est le décalage vers 5 qui est fait d'après ce qui est décrite dans la table action.

0	E	1	+	5

Figure 50 Etat 9 de la pile

La chaîne qui n'est pas encore analysée est id*id$.La pile montre qu'on est en 5, on veut lire id, la table action nous montre que id va être lu, on fait le décalage vers 3. Le contenu de la pile est représenté par la figure 51.

0	E	1	+	5	id	3

Figure 51 Etat 10 de la pile

Etant dans 3 et qu'on veut lire *, on applique la réduction selon la table action, id est alors remplacé par F.

0	E	1	+	5	F

Figure 52 Etat 11 de la pile

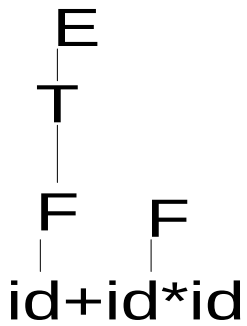


Figure 53 Réduction de id en F

La table successeur indique que 5 et E donnent 2.

0	E	1	+	5	F	2

Figure 54 Etat 12 de la pile

Etant alors en 2 en lisant *, c'est la réduction qui est appliquée F est remplacé par T.

0	E	1	+	5	T

Figure 55 Etat 13 de la pile

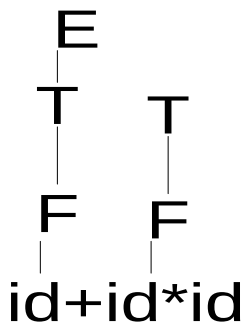


Figure 56 Réduction de F en T

D'après la table Successeur 5 et T donnent 8.

0	E	1	+	5	T	8

Figure 57 Etat 14 de la pile

Etant dans 8,* est lu en faisant le décalage selon la table action.

0	E	1	+	5	T	8	*	6

Figure 58 Etat 15 de la pile

La chaîne qui n'est pas encore analysée est id$, on procède son analyse comme suit. Etant dans 6 pour lire id, un décalage est appliqué. Id est lu et on décale vers 3.

0	E	1	+	5	T	8	*	6	id	3

Figure 59 Etat 16 de la pile

La chaîne qui n’est pas encore analysée est $.Etant dans 3 lire $ revient à faire la réduction selon la table action, id est remplacé par F .

0	E	1	+	5	T	8	*	6	F

Figure 60 Etat 17 de la pile

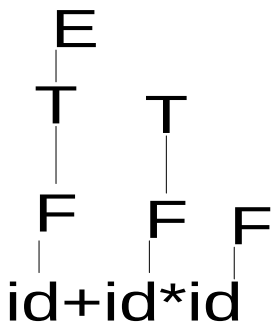


Figure 61 Réduction de F en T

Dans la table de successeur 6 et F donnent 7.

0	E	1	+	5	T	8	*	6	F	7

Figure 62 Etat 18 de la pile

Maintenant c’est le caractère $ qui est concerné par la lecture .Etant dans 7 lire $ revient à faire la réduction selon la production respective, T*F sont remplacés par T .

0	E	1	+	5	T

Figure 63 Etat 19 de la pile

L’arbre syntaxique devient

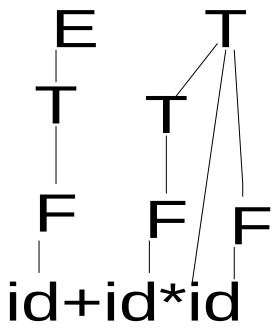


Figure 64 Réduction de T*F en T

Etant dans 5 et comme non terminal T, la table successeur montre qu'on a 8.

0	E	1	+	5	T	8

Figure 65 Etat 20 de la pile

Lire $ étant dans 8, la table action nous montre qu'il faut faire la réduction en remplaçant E+T par E.

0	E

Figure 66 Etat 21 de la pile

L'arbre syntaxique correspondant est représenté à la figure 67.

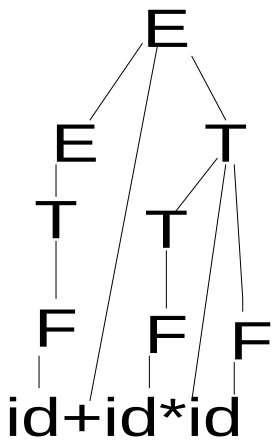


Figure 67 Réduction de E+T en E

Etant à 0 et un non terminal E la table de successeur nous montre qu'on avance vers 1.

0	E	1

Figure 68 Etat 22 de la pile

En lisant $ étant dans 1, la table action nous montre accepter comme action.la chaîne est alors engendrée par la grammaire, l ‘arbre syntaxique est représenté à la figure 69.

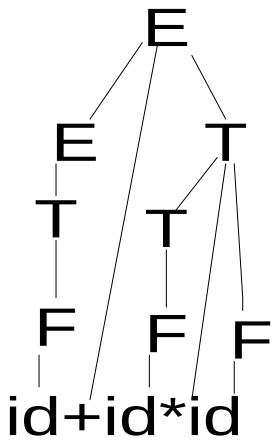


Figure 69 Arbre syntaxique obtenue par SRL

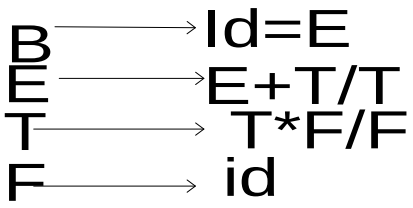
IV.4.Analyse sémantique

Le but de l'analyse sémantique est de vérifier la présence d’erreurs d’ordre sémantique (vérification de type).Utilisation de l’arbre résultat de l’analyse syntaxique.[18]

IV.4.1 Arbre syntaxique décoré et l’arbre syntaxique abstrait

L’arbre syntaxique ou arbre de dérivation d’un mot du langage généré par une grammaire est souvent inutilement complexe. Lorsqu’on génère un arbre lors de l’analyse syntaxique, on préfère une représentation condensée appelée arbre abstrait. Dans celui-ci, les mots-clés sont souvent supprimés et les opérateurs remontés sur le nœud père [19].

La construction de l’arbre syntaxique abstrait doit être faite pendant cette phase de l’analyse sémantique. C’est un résumé de l’arbre syntaxique où les éléments utiles pour la suite sont seulement conservés sur cette représentation. Durant cette phase, l’arbre syntaxique décoré est dressé en utilisant des grammaires attribuées. Considérons le mot h=t*h+s extrait de notre programme et la grammaire suivante :



Pendant la phase d'analyse syntaxique le mot h=t+h*s est représenté comme montre la figure 70.

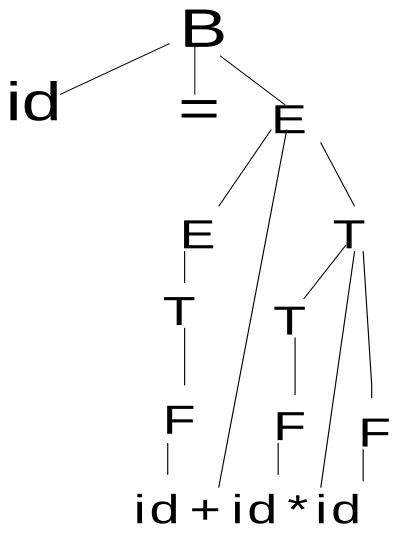


Figure 70 Arbre syntaxique du mot

L'arbre syntaxique abstrait correspondant est la figure 71.

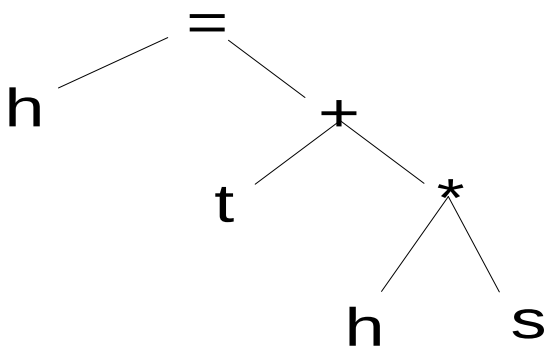


Figure 71 Arbre syntaxique abstrait du mot analysé

IV.4.2 Traduction dirigée par la syntaxe

La traduction dirigée par la syntaxe est réalisée en attachant des actions sémantiques aux règles de la grammaire. Elle repose sur deux concepts :

-Les attributs. Un attribut est une quantité quelconque associée à une construction du langage de programmation.

- Les schémas de traduction sont une notation permettant d'attacher des fragments de programme aux règles de la grammaire.[20] La syntaxe est enrichie par des attributs et les règles d'attribution. [21] Des attributs sont associés à chaque symbole de la grammaire et les actions sémantiques sont associées à chaque production pour déterminer la valeur des attributs associés. La traduction dirigée par la syntaxe est constituée par l'ensemble des actions sémantiques et la grammaire. Les attributs sont groupés dans deux catégories : les attributs synthétisés et les attributs hérités .Un attribut est synthétisé si sa valeur à un nœud d'un arbre syntaxique est déterminée à partir des valeurs d'attributs des fils de ce nœud. L'attribut de la partie gauche est calculé en fonction des attributs de la partie droite. Un attribut hérité est un attribut dont la valeur à un nœud d'un arbre syntaxique est déterminée en fonction des attributs du père et/ou des frères de ce nœud. Les calculs des attributs sont effectués à partir du non terminal de la partie gauche. Un arbre syntaxique sur les nœuds duquel on rajoute la valeur de chaque attribut est appelé arbre syntaxique décoré. Dans notre travail avec cette grammaire ci-dessous, la traduction dirigée par la syntaxe, et l'arbre syntaxique décoré sont illustrés dans le tableau 15.

$$
\begin{aligned}
B &\longrightarrow Id=E \\
E &\longrightarrow E+T/T \\
T &\longrightarrow T*F/F \\
F &\longrightarrow id
\end{aligned}
$$

Tableau 15 Attribut valeur

Production	Actions sémantiques
B⟶id=E	B.valeur=id.valeur=E.valeur
E⟶E+T	E.valeur=E.valeur+T.valeur
E⟶T	E.valeur=T.valeur

T⟶T*F	T.valeur=T.valeur*F.valeur
T⟶F	T.valeur=F.valeur
F⟶id	F.valeur=id.valeur

Pour les attributs synthétisés les valeurs sont calculées à partir des feuilles vers l'axiome. L'arbre décoré correspondant à cette Traduction dirigée par la syntaxe avec des valeurs,4,7,5 qui peuvent être affectées aux identifiants est représenté comme indiqué sur la figure 72.

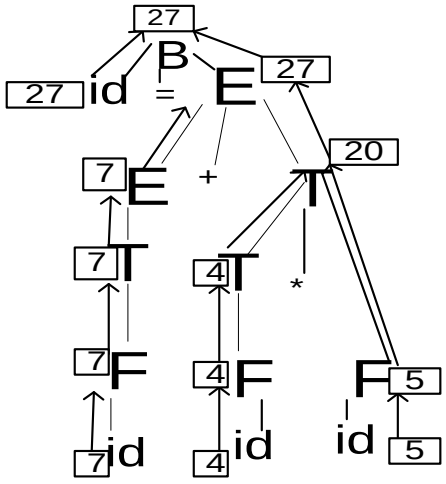


Figure 72 Attributs synthétisés

Il est illustré dans le tableau suivant, l'attribut type qui est considéré comme attribut hérité.

Tableau 16 Attribut type

Production	Actions sémantiques
B⟶id=E	B.type=id.type=E.type
E⟶E+T	E.type=E.type+T.type
E⟶T	E.type=T.type

T⟶T*F	T.type=T.type*F.type
T⟶F	T.type=F.type
F⟶id	F.type=id.type

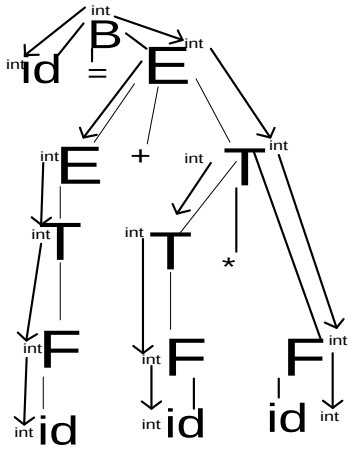


Figure 73 Attribut hérité

IV.5. Phase de synthèse

Pendant cette phase le code cible est généré à partir de la représentation intermédiaire produite à la fin de la phase d'analyse. Cette représentation intermédiaire peut être sous forme d'arbre syntaxique abstrait ou sous forme de code en trois adresses.

IV.5.1 Génération du code à partir du code en trois adresses.

La représentation sous forme de code en trois adresses est constituée par une suite d'instructions dont chacune a au plus trois opérandes. Pour ce cas la génération de code cible est effectuée en trois étapes qui sont les suivantes :

-la génération de code intermédiaire

- L'optimisation de code intermédiaire

-La génération de code cible

La génération de code intermédiaire traduit l'arbre de syntaxe abstraite en un code pour une machine abstraite [22] Traduction dirigée par la syntaxe peut être utilisée pour générer le code intermédiaire. Avec la grammaire suivante le code intermédiaire du mot h=t+h*s peut être généré.

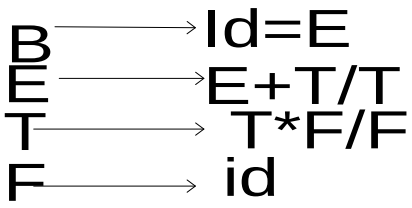


Tableau 17 Production et actions sémantiques correspondantes

Production	Actions sémantiques
B⟶id=E	Generer(id.nom=E.place)
E⟶E+T	E.place=new temp() ;generer(E.place=E.place+T.place
E⟶T	E.place=T.place
T⟶T*F	T.place=new temp() ;generer(T.place=T.place*F.place
T⟶F	T.place=F.place
F⟶id	F.place=id.nom

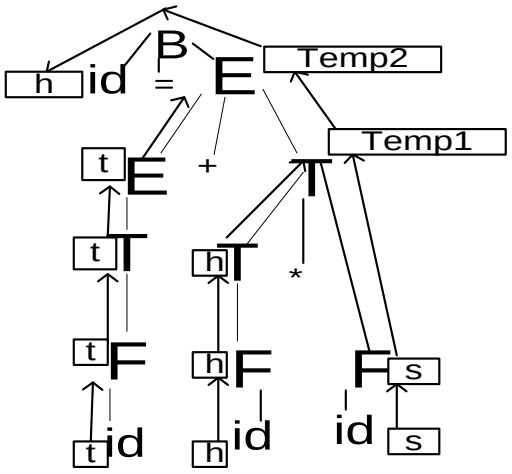


Figure 74 Arbre syntaxique décoré

Temp1= h*s

Temp2=t+temp2

h=temp2

L'optimisation du code intermédiaire est réalisée pour que l'exécution soit la plus rapide que possible. En optimisant le code, le compilateur commence par réaliser un travail sur le code intermédiaire en diminuant la taille du code (travail sur les boucles : réduction du nombre de variables et du nombre d'opérations). [23] Le code optimisé issu du code intermédiaire est le suivant :

Temp1= h*s

h=t+temp1

Le compilateur produit du code en langage d'assemblage, qui est ensuite traduit par un assembleur en code machine. [24] Pour notre cas le code en langage d'assemblage du code optimisé précédemment est le suivant.

LOAD@s,R0;

LOAD@h,R1;

MUL R1, R0;

STORE R0,temp1 ;

LOAD@t,R0;

ADD temp1,R0 ;

STORE R0,@h ;

IV.5.2.Génération du code à partir de l'arbre syntaxique abstrait.

Considérons l'arbre abstrait de la figure 75 qui est généré au niveau de l'analyse sémantique.

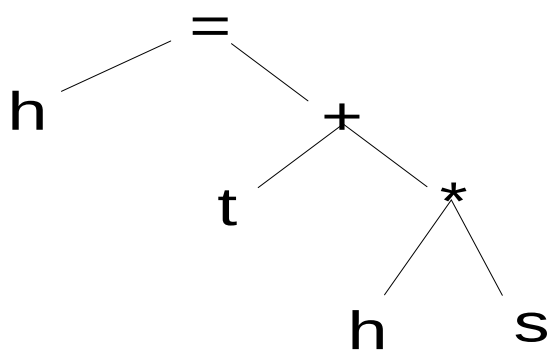


Figure 75 Arbre syntaxique abstrait pour la génération du code

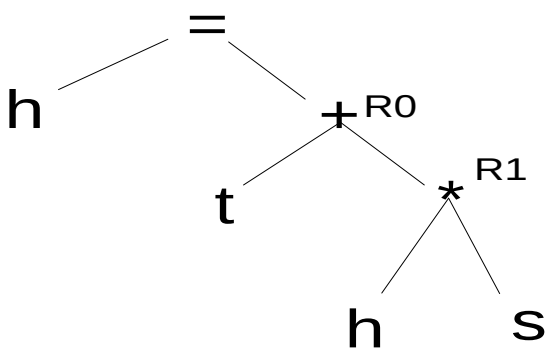


Figure 76 Manipulation des variables par des registres

LOAD@s,R1;

MUL @h, R1;

LOAD@t,R0;

ADD R1, R0;

STORE R0, @h;

IV.6.Conclusion

Pour bien maîtriser la carrière de développement des logiciels, il est nécessaire de savoir le fonctionnement du compilateur puisqu'on pourrait donner naissance à un compilateur nouveau dédié à des tâches quelconques spécifique des clients.

CONCLUSION GENERALE ET RECOMMANDATIONS

CONCLUSION GENERALE

Notre travail concerne l'étude de la compilation de l'implémentation de l'algorithme du cryptosystème de diffie-hellman et il est constitué de 4 chapitres.

Dans le premier chapitre qui concerne l'introduction générale, nous avons expliqué l'importance de notre projet vis-à-vis des entreprises.

Dans le deuxième chapitre, les quatre systèmes cryptographiques avec échange sécurisé des clés sont explorés. Les systèmes cryptographiques avec échange de clé sécurisé sont de grande importance puisque la clé est échangée entre les interlocuteurs sans que l'espion aperçoive la clé .Cet échange est effectué dans les meilleurs délais contrairement à l'échange de clé pour les cryptosystèmes asymétriques.

Le troisième chapitre est dédié à l'implémentation du cryptosystème de diffie-hellman. Cet algorithme consiste à la génération des clés. L'importance de cet algorithme réside à la difficulté de résoudre un problème du logarithme discret dans certains groupes et surtout si les nombres utilisés sont grands comme ceux de 512 bits, 1024bits ou 2048bits voire plus. L 'application du cryptosystème de diffie-hellman pouvant être heurtée à l'attaque dite man in the middle, une signature numérique est utilisée pour pallier à cette attaque. La fonction de hachage et le cryptosystème RSA sont utilisés comme signature numérique.

Le quatrième chapitre est dédié à l'étude de la compilation proprement dite. En premier lieu, pendant la phase d'analyse quelques exemples d'instruction de notre programme sont analysés en utilisant un automate à état fini pendant l'analyse lexicale. Des algorithmes comme LL(1) pour l'analyse syntaxique descendante et les algorithmes LR(0), SLR(0) sont utilisés pour l'analyse syntaxique ascendante pendant la phase d'analyse syntaxique. Pendant cette phase nous avons montré avec des exemples à l'appui tirés de notre programme comment une suite des mots peut être acceptée par une grammaire s'il appartient au langage. L'arbre syntaxique est aussi dressé. Cet arbre syntaxique est traité pendant l'analyse sémantique en combinaison avec les attributs pour vérifier le sens. L'arbre syntaxique abstrait de la phase d'analyse sémantique est aussi obtenu à partir de cet arbre syntaxique. Cet arbre

syntaxique abstrait est ensuite utilisé pour générer le code final pendant la phase de synthèse. Un code cible est généré en assembleur.

Recommandations

Des recommandations sont adressées aux futurs lauréats et aux entreprises utilisant le compilateur et aux entreprises qui intègre le système de chiffrement.

Recommandations adressées aux entreprises

-Les développeurs doivent approfondir le fonctionnement d'un compilateur.

-Les entreprises devraient s'équiper des outils nécessaires à la cryptographie quantique.

Recommandations aux futurs lauréats

Pour futurs lauréats qui sont intéressés par la programmation et la cryptographie ; ils sont appelés à approfondir:

- Les notions de la physique quantique et les outils à utiliser pour la mise en jeu de la cryptographie quantique.

Le fonctionnement d'un compilateur qui leur permettra de bien développer des logiciels.

-Les systèmes cryptographiques avec échange sécurisé des clés.

REFERENCES

OUVRAGES

[1] S .Pasini,les versions theoriques d'ElGamal,Mars 2005

[2] R.Alléaume,Cryptographie quantique,Ecole Nationale Supérieure des Télécommunications-Paris,17

[3] Cécile GONÇALVES,Cryptographie Avancée, Université Paris-Est

[4] Cécile GONÇALVES,Cryptographie Avancée, Université Paris-Est

[5] Adderrahmane NITAJ ,Université de Caen,France Rabat 29 Octobre 2008

[6] Théorie des langages et de la compilation,Thierry Massart,

[7] Cédric Bastoul,introduction à la compilation,Université de Strasbourg

[8] Jean Méhat,Interprétation et compilation,Université de Paris 8 Vincennes Saint Denis,

[9] H.Fouchal,Langages et Compilation,Université de Reims Champagne-Ardenne

[10] Prof. M.D. RAHMANI,Cours de Compilation,Université Mohammed V – Agdal

[11] A KHOUMSI, ORGANISATION DES LANGAGES ET COMPILATION,Université de Sherbrooke,Hiver 2002

[12] Jean-Guy Mailly,Theorie des langages, Université Paris Descartes

[13] Claude M,Théorie des Langages, Université de Technologie de Compiègne,Printemps

[14] Anne BERRY,Support de cours de Compilation,M1 Informatique,2013-2014

[15] Jacques Farré,Compilation, Université de Nice

[16] E. RAMAT,Compilation, Université de Technologie de Compiègne,Mars 2009

[17] Claude M,Théoriedes Langages,Université de Technologie de Compiègne,Printemps

[18] Michaël Krajecki,Langages et compilation,Université de Reims Champagne-Ardenne

[19] Michel Meynard,Interprétation et Compilation,14 janvier 2016

[20] Alexis Nasr,Carlos Ramisch,Sylvain Sené,Aix Marseille Université

[21] Martin Odersky,Grammaires Attribuées,16 décembre 2005

[22] Cédric Bastoul,introduction à la compilation,Université de Strasbourg

[23] Romaric DAVID,Compilation,LEM2I - Décembre 2011.

[23] Abdelmajid Dargham,Introduction `a la compilation, Université Mohamed Premier,Septembre,2012

WEBOGRAPHIE

[2 .1] https://www.securiteinfo.com cryptographie visité le 12 septembre 2018.

[2.2] http://www.lkb.upmc.fr/quantumoptics visité le 5 Juillet 2018

[2.3] https://pastel.archives-ouvertes.fr/pastel-00003416 visité le 5 Juillet 2018

[2.4] http://www.ictea.com/cs/knowledgebase.php visité le 20 septembre, 2018.