الجـــــــــــــمـهـوريـــــة الجـــــــــــــزائـريـة الديمـــــــقـراطيـة الـشـــــــــعبيـة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التـــــــــــــــــعـليـم العـــــــالي والبـــــــــحث العـــــلـمـــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جـــــــــــــــامعة أبي بكـــر بـلـقـايد – تـلمســـــــــــــــان –

Université Aboubakr Belkaïd– Tlemcen –
Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications
Spécialité : Réseaux et Télécommunications

Par : BOUDGHENE STAMBOULI Kawter
BOUDJEMAA Chaïmaa

Thème

Sécurisation d'un réseau bancaire avec la technologie Blockchain

Soutenu publiquement, le 26 / 08 / 2020, devant le jury composé de :

Mr. MERZOUGUI Rachid	PR	Univ. Tlemcen	Président
Mr. HADJILA Mourad	MCA	Univ. Tlemcen	Directeur de mémoire
Mr. BENMOUSSAT Chemseddine	MCB	Univ. Temouchent	Co-Directeur de mémoire
Mr. IRID Sidi Mouhamed	MCA	Univ. Tlemcen	Examinateur

Année Universitaire 2019-2020

Dédicaces

Je dédie ce travail

A ma famille, elle qui m'a doté d'une éducation digne, son amour a fait de moi ce que je suis aujourd'hui. Je suis très fière d'être votre fille et de pouvoir enfin réaliser, ce que vous avez tant espéré et attendu de moi.

Mon adorable mère Soufi Merzougue Faiza Quoi que je fasse ou que je dise, je ne saurai point te remercier comme il se doit. Ton affection me couvre, ta bienveillance me guide et ta présence à mes côtés a toujours été ma source de force pour affronter les différents obstacles.

A mes très chers frères Younes et Lokmane Rezk Ellah. Puisse Dieu vous donne santé, bonheur, courage et surtout réussite. J'espère que nous resterons toujours aussi unis.

A mon adorable petite sœur Marwa Firdawse qui sait toujours comment procurer la joie et le bonheur pour toute la famille.

A ma chère tante Kara Ali Amina cela n'a pas cessé de me conseiller encourager et soutenir tout au long de mes études. Que Dieu la protège et leur offre la chance et le bonheur.

A mon cher grand-père, Qui je le souhaite une bonne santé. Ainsi, à ma grand-mère que dieu vous grade.

*A ma défunte grand-mère qui disait souhaité voir de là où elle est réussie mes études, qu'**ALAHE** l'accueille dans son vaste paradis. Si Dieu a mis le paradis sous les pieds des mères, ce n'est pas pour rien. Tu représentes la source de tendresse et l'exemple du dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Ta prière et ta bénédiction m'ont été d'un grand secours pour mener à bien mes études. Je t'aime énormément.*

A ma confidente : Mlle Chaima Oualichaouche Conserve-moi ta profonde amitié et ton immense amour et sois convaincue qu'il en est de même pour moi.

A Mlle Nadjet Bouchenafa et Mlle Riham Boudghene Stambouli vous êtes la plus belle rencontre scientifique et amicale. Vos conseils et encouragements m'ont donné du tonus pour aller de l'avant. Je sais que tu es là, jamais je ne t'oublierai puisque je te dois beaucoup d'affection et amour !

Sans oublier mon binôme Chaimaa pour son soutien moral, sa patience et sa compréhension tout au long de ce projet.

Mlle. Boudghene Stambouli Kawter

Dédicaces

Je dédie ce travail

Avec l'expression de ma reconnaissance, je dédie ce modeste travail à ceux qui, quels que soient les termes embrassés, je n'arriverais jamais à leur exprimer mon amour sincère.

A mon cher père Mouhamed, Permettez-moi de vous exprimer mon grand amour mon attachement et ma plus haute considération pour Votre personne. Cher père, veillez trouver, dans ce modeste travail, le fruit de vos sacrifices ainsi que l'expression de ma profonde affection et ma vive reconnaissance Que Dieu vous protège et vous garde.

Ma Chère Maman, Si Dieu a mis le paradis sous les pieds des mères, ce n'est pas pour rien. Affable, honorable, aimable : Tu représentes pour moi le symbole de la bonté par excellence. Aucune dédicace ne saurait être assez éloquente pour exprimer ce que tu mérites pour tous les sacrifices que tu n'as cessé de me donner depuis ma naissance, durant mon enfance et même à l'âge adulte. Tu as fait plus qu'une mère puisse faire pour que ses enfants suivent le bon chemin dans leur vie et leurs études. Je prie Dieu, le tout-puissant, de l'accueillir en son vaste paradis. Amen.

A ma chère sœur Karima et son marie Mouhamed, et son petit adorable Ayoub. Je vous souhaite la réussite dans votre vie, avec tout le bonheur qu'il faut pour vous combler. J'ai de la chance de t'avoir. Que Dieu vous garde.

A ma petite sœur Wafaa, Je te dédie pour tous les moments de joie et de taquineries qu'on a passé ensemble. Je prie dieu, le tout puissant de t'accorder santé, bonheur et succès…adorable SŒUR !

Mon cher frère Nassreddin, Ambitieux comme tu es, je te souhaite beaucoup de succès et de bonheur pour couronner ton courage et tes sacrifices Puisse Dieu t'accorder la bonne chance et la réussite dans tes études. J'espère que nous resterons toujours aussi unis.

Au Dr.Ghomari Meriem l'épouse de mon père qui ma aidé et supporté dans les moments difficiles.

A celui que j'aime beaucoup et qui m'a soutenue tout au long de ce projet, ma chère amie Fatima Medjahdi, et bien sur mes tantes Amel Khebichat et Soumia Boussaidi sans oublier mon oncle Fouzi Boudjemaa.

A toute la famille Boudjemaa, et à ma cher binôme Kawter pour son soutien moral, sa patience et sa compréhension tout au long de ce projet.

Mlle. Boudjemaa Chaimaa

Remerciement

*Tout d'abord nous adressons nos plus sincères sentiments de reconnaissance et de remerciement envers **ALLAH**, le clément et le miséricordieux, lequel nous a accordé la force et le courage de mener bien ce modeste travail.*

*Notre gratitude s'adresse à Monsieur **Hadjila Mourad** Professeur à l'université de Tlemcen pour son encadrement, son orientation, ses conseils et la disponibilité qu'il nous a témoignée pour nous permettre de mener à bien ce travail. Ses conseils et son support moral nos énormément aidé à mener à terme ce travail.*

*Nos vifs remerciements à notre Co-encadreur, Monsieur **Benmoussat ChemsEddin**, Professeur à l'université de Ain-Temouchent, pour nous avoir codirigé, soutenu, encouragé et orienté tout au long de ce projet.*

Nos vifs remerciements aux membres du jury d'avoir accepté d'examiner et d'évaluer notre travail. Un grand merci à tous les professeurs de Télécommunications qui ont participé à notre progrès pendant ces 5 ans.

Enfin, nos remerciements à tous nos amis, nos collègues qui nous ont soutenu et encouragé pour la réalisation de cet humble mémoire.

Résumé

Notre monde change et évolue de plus en plus au rythme des innovations et des nouvelles technologies. La blockchain a été rendue populaire à la suite d'un article que le magazine The Economist lui a consacré fin 2015. Nous entendons de plus en plus parler dans la presse du bitcoin, une monnaie virtuelle basée sur la technologie blockchain. Ce projet de fin d'études vise à vulgariser et démystifier, de manière pédagogique, le fonctionnement de la blockchain. Nous en expliquerons les grandes lignes sans avoir besoin d'être un expert en informatique pour comprendre. Cette technologie a le potentiel de changer les règles du jeu pour de nombreux secteurs, y compris le secteur financier, à commencer par le système bancaire. Ce nouveau modèle de confiance basé sur un système décentralisé et partagé veut remplacer l'intermédiaire de confiance existant. Un nouveau monde émerge. Dans ce travail, nous nous intéressons à l'utilisation de la Blockchain pour la gestion des opérations sur les transactions bancaires. La proposition donne solution aux différents problèmes liés au domaine du vol et fraude de la monnaie en se basant sur des transactions similaires à la transaction du bitcoin.

Mots clés : Blockchain, cryptocurrency, bitcoin, ethereum, litcoin, transaction, sécurité informatique, réseau bancaire, réseau personnel, réseau local, les réseaux métropolitains, cryptographie, adressage IP.

Abstract

Our world is changing and evolving more and more at the pace of innovations and new technologies. The blockchain was made popular following an article that The Economist magazine devoted to it at the end of 2015. We hear more and more in the press about bitcoin, a virtual currency based on blockchain technology. This end of studies project aims to popularize and demystify, in an educational way, the functioning of the blockchain. We will explain the main lines without having to be a computer expert to understand. This technology has the potential to change the rules of the game for many sectors, including the financial sector, starting with the banking system. This new trust model based on a decentralized and shared system wants to replace the existing trust intermediary. A new world is emerging. In this work we were interested in the use of Blockchain for the management of operations on bank transactions. The proposal provides a solution to the various problems related to the theft and fraud of money based on transactions similar to the bitcoin transaction.

Keyword: Blockchain, cryptocurrency, bitcoin, ethereum, litcoin, transaction, computer security, banking network, personal network, local network, metropolitan networks, cryptography, IP addressing.

ملخص

إن عالمنا يتغير ويتطور أكثر فأكثر بوتيرة الابتكارات والتكنولوجيات الجديدة. تم نشر تقنيةblockchain بعد مقال خصصته مجلة The Economist في نهاية عام 2015. نسمع المزيد والمزيد في المجلات حولbitcoin ، وهي عملة افتراضية تعتمد على هذه التقنية. والتي كانت أول تطبيق لتقنية blockchain في عام 2009. يهدف هذا المشروع النهائي للدراسة إلى تعميم طريقة عمل blockchain وإزالة الغموض عليها بطريقة تعليمية. سنشرح النقاط الرئيسية دون الحاجة إلى أن نكون خبراء كمبيوتر لفهم هذه التكنولوجيا التي لديها القدرة على تغيير قواعد اللعبة للعديد من القطاعات بما فيها القطاع المالي، بدءا من النظام المصرفي. يريد هذا القطاع استبدال وسيط الثقة الحالي المستند مع النظام اللامركزي.

في هذا العمل، كنا مهتمين باستخدام Blockchain لإدارة العمليات على المعاملات المصرفية حيث نقدم اقتراحا لحل هذه المشكلة المتعلقة بمجال السرقة والاحتيال التي بنيت على المعاملات المشابهة لمعاملة bitcoin

Table des matières

Liste des tables

Liste des équations

Liste des figures

Liste des abréviations

A

AES Advanced Encryption Standard

B

BCH Bitcoin Cash

BGP Border Gateway Protocol

BPFT Pratical Byzantine Fault Tolerant

BTC BitCoin

BC BlockChain

C

CIDR Classless Inter-Domain Routing

CPU Central Processing Unit

D

DES Data Encryption Standard

DHCP Dynamic Host Configuration Protocol

DNS Domain Name Service

DSA Digital Signature Algorithm

DSS Digital Signature Standard

DLT Distributed Ledger Technologies

E

ECDSA Elliptic Curve Digital Signature Algorithm

EGP Exterior Gateway Protocol

EIGRP Enhanced Interior Gateway Routing Protocol

ETH Ethereum

F

FIPS Federal Information Processing Standards

FTP File Transfer Protocol

G

GPU Graphics Processing Unit

H

HUB Housing Union Building

HTTP Hypertext Transfer Protocol

I

IBM International Business Machines

ICMP Internet Control Message Protocol

IGRP Interior Gateway Routing Protocol

IGP Interior Gateway Protocol

IP Internet Protocol

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

IPsec Internet Protocol Security

IS-IS Intermediate System to Intermediate System

ISP Internet Service Provider

ISO International Organization for Standardization

IKEv2 Internet Key Exchang Version 2

ID Identificateur

J

K

L

LAN Local Area Network

LLC Logical Link Control

L2TP Layer 2 Tunneling Protocol

LTC LiteCoin

M

MAN Metropolitan Area Network

MAC Message Authentication Code

MD5 Message Digest 5

MySQL : est une combinaison de "My", le nom de la fille du co-fondateur Michael Widenius, et "SQL", l'abréviation de StructuredQueryLanguage.

N

NFC Near Field Communication

NIST National Institute of standards and Technology

NSA National Security Agency

NEO Non-Equity Option

NIC Network Interface Card

O

OSIOpen Systems Interconnection

OSPF Open Shortest Path First

P

PAN Personal Area Network

PBFT Practical Byzantine Fault Tolerance

PoW Proof of Work

PPP Point-to-Point

P2P Peer-to-Peer

POS Proof of Stacke

POA Proof of Authority

OSPF Open Shortest Path First

Q

R

RC4 Rivest Cipher 4

RIP Routing Information Protocol

RSA Rivest–Shamir–Adleman

REP Repons

S

SHA Secure Hash Algorithm

SSH Secure Socket Shell

SSTPSecure Socket Tunneling Protocol

T

TCP/IP Transmission Control Protocol/Internet Protocol

TDES Triple Data Encryption Standard

U

UDP User Datagram Protocol

V

VPN Virtual Private Network

W

WAN Wide Area Network

WEB world wide web

X

XOR exclusive O

Y

Z

Introduction générale

La blockchain est un terme dans le domaine des technologies de l'information. Il s'agit essentiellement d'une base de données partagée et les données ou informations qui y sont stockées ont les caractéristiques "infalsifiable", "trace laissée", "traçable", "ouverte et transparente" et "maintenance collective". Sur la base de ces caractéristiques, la technologie blockchain a jeté des bases solides de « confiance », créant un mécanisme de coopération fiable et ayant de larges perspectives d'application.

Le 10 janvier 2019, le National Internet Information Office a publié le « Règlement sur la gestion des services d'information sur la blockchain ». Le 24 octobre 2019, lors de la dix-huitième étude collective du Bureau politique du Comité central, le secrétaire général Xi Jinping a souligné que "prendre la blockchain comme une percée importante dans l'innovation indépendante des technologies de base" "accélérer le développement de la technologie blockchain et l'innovation industrielle". La « blockchain » est entrée dans le champ de vision du public est devenue le centre d'intérêt de la société.

La blockchain provient du Bitcoin de Satoshi Nakamoto. En tant que technologie, la blockchain est une solution technique qui ne repose pas sur des tiers et utilise ses propres nœuds décentralisés pour stocker, vérifier, transférer et communiquer les données du réseau. Par conséquent, c'était sous un angle de comptabilité financière, la technique blockchain est considérée comme un grand centre de facturation de réseau distribué. Tout le monde peut utiliser en même temps les normes techniques, ajoute leurs propres informations, étend La blockchain et continue de répondre aux besoins de saisie de données induites par divers besoins. Cette technologie, qui fait l'objet de notre travail, a été adaptée avec succès sur l'étude et la conception d'un réseau bancaire sécurisée.

Le manuscrit est divisé en quatre chapitres :

Le premier chapitre est consacré tout d'abord à la présentation générale de l'ouvrage dont nous avons couvert l'évolution de la blockchain, son histoire et sa définition. Ensuite, nous présenterons les avantages de la conception et le fonctionnement de cette technologie passant par la présentation de l'importance de la blockchain suivie par certains cas d'utilisation. Nous définirons également les types de la blockchain, et nous conclurons par les mécanismes de sécurité dans les réseaux.

Le deuxième chapitre présente l'étude théorique de la technologie blockchain pour l'étude et la conception d'un réseau bancaire sécurisé avec la blockchain. Dans cette étude, nous allons détailler le concept de crypto-monnaies qui permettent les paiements sécurisés en se basant sur plusieurs méthodes de chiffrements. Nous définirons également, les divers algorithmes de cryptage et techniques cryptographiques qui protègent les informations, tels que le cryptage à courbe elliptique, les paires de clés public-privée et les fonctions de hachage. Ensuite et dans le but de valider la compréhension de la blockchain, nous avons étudié le procédé technique sur lequel repose cette technologie.

Dans le troisième chapitre, nous allons développer les réseaux informatiques qui consistent de créer un réseau bancaire. Nous couvrirons les types des réseaux tels que PAN, MAN, WAN, ect. Pour cela, nous allons étudier de façon approfondie l'adressage IP. Par la suite,

nous donnerons une vue globale sur le modèle OSI et TCP/IP, ainsi les notions de protocole et routage.
Dans le quatrième chapitre, nous exposerons une nouvelle technique qui est basée sur la combinaison entre la technologie blockchain et le réseau bancaire. Cette technique nous permettra de construire un réseau bancaire avec un routage amélioré, sécurisé et rendant le trafic plus efficace. Pour répondre à ce besoin de communication sécurisé, nous avons utilisé un réseau privé virtuel (VPN) avec ses différentes topologies. Dans ce cadre, nous allons présenter la fonction constructrice, puis nous sommes passés à la création de méthodes étonnantes telles que creatNewBlock, creat NewTransaction, getLastBlock, etc. Nous proposerons, par la suite, la méthode de hachage, le hachage SHA256, et nous créerons une méthode pour générer un hachage pour nos données de bloc. Nous avons également appris ce qu'est une preuve de travail et comment cela fonctionne. De même, nous configurerons Express.js dans notre projet, et verrons comment utilisez-le pour construire notre API / serveur. Nous conclurons par la création d'un réseau décentralisé.

Chapitre I : Généralités et présentation de l'ouvrage

I.1 Historique

Au début des années 1990, deux crypto-logues Américains Stuart Haber et W. Scott Stornetta imaginent un système informatique que tout le monde peut écrire mais qui est impossible à effacer et indestructible. Puis dans les années 1991, les chercheurs ont trouvé une solution qui utilise des techniques de sécurité cryptographique pour garantir que tous les fichiers soient sécurisés par la Blockchain. Plus tard en 1992, la technologie de MarkleTree également été intégrée du système, permettant de stocker simultanément plusieurs documents dans le même bloc. Il est dommage que cette technologie ait ensuite été ignorée et lentement abandonnée. Malheureusement, ce brevet a été annulé en 2004 parce qu'il n'avait pas été mis à jour, quatre ans seulement avant le boom des Bitcoins. En 2004, l'informaticien et activiste en cryptographie Hal Finney (Harold Thomas Finney II) a présenté un système de preuve de travail réutilisable appelé PoW (proof of work). Le système reçoit des jetons de travail hashcash[2] irremplaçables, qui créent en retour des pièces pouvant être transférées entre utilisateurs et signées par RSA.[1]

PoW peut être considéré comme un premier prototype de la blockchain, constituant une première étape importante dans le cryptage de l'historique des devises.

La fin de l'année 2008 a vu la publication d'un livre blanc sur un système de paiement électronique décentralisé poste à poste appelé Bitcoin sur une liste de diffusion cryptée par des individus ou des groupes sous le pseudonyme Satoshi Nakamoto. Bitcoin est basé sur le calcul de la preuve de travail Hascash[2]. Par rapport à l'opération très fiable POW, Bitcoin utilise un protocole décentralisé de point à point pour vérifier et suivre les transactions afin d'éviter une double consommation. En bref, Bitcoin "exploite" en utilisant le mécanisme de contrôle du travail de chaque mineur, puis vérifié par des nœuds décentralisés du réseau *[2]*.

Le 3 janvier 2009, Satoshi Nakamoto a extrait et exploité le premier bloc de Bitcoins. Il a également reçu un bonus de 50 Bitcoins. Le 12 janvier 2009, le premier destinataire de Bitcoin était Hal Finney, qui avait reçu 10 Bitcoins de Satoshi Nakamoto. Il s'agissait de la première transaction confirmée par Bitcoin au monde *[1]*.

En 2013, le programmeur et co-fondateur de Magazine Bitcoin, VitalikButerin, a déclaré que Bitcoin devait écrire un langage de script pour créer des applications décentralisées. Ne parvenant pas à obtenir un accord au sein de la communauté, Vitalik a commencé à développer une nouvelle plate-forme d'informatique répartie basée sur une chaîne de blocs, Ethereum, dotée de fonctions de script, également appelées contrats intelligents *[3]*.

---

[1] **RSA :** est l'un des premiers crypto systèmes à clé publique et est largement utilisé pour la transmission sécurisée de données

[2] **Hashcash :** est un algorithme de preuve de travail, qui a été utilisé comme technique de contre-mesure de déni de service dans un certain nombre de systèmes.

I.2 Définition

Le monde change continuellement, entraine par les innovations technologiques qui affectent notre façon de vivre et de faire des affaires. Cependant, si les technologies de l'information et de la communication ont profondément affecté l'organisation de la production, elles ne sont pas encore réussies à numériser les transactions[3] commerciales. Malgré les efforts déployés récemment pour mettre en place des processus électroniques destinés à traiter certains aspects des procédures commerciales, les transactions commerciales restent encore fortement dépendantes du papier. Les problèmes de sécurité surgissent sur les flux de données à travers les frontières. Une nouvelle technologie, Blockchain, est considérée comme un changeur de jeu possible avec son mécanisme unique de renforcement de la confiance, est devenue une direction importante pour l'intégration en profondeur des finances et de la technologie. Mais qu'est-ce que la Blockchain et quel est le potentiel de cette technologie pour le commerce international ? *[4]*

La Blockchain est partout en ce moment. Mais tout le monde n'a pas encore bien compris de quoi on parlait, la phase d'évangélisation n'est pas encore finie, voici une explication assez courte qui peut vous aidez à comprendre de quoi il s'agit et aussi à l'expliquer aux autres :

Imaginez un grand livre ou un enregistrement numérique de transaction que tous peuvent lire gratuitement et librement, ce livre est décentralisé, aucune entité ne contrôle le réseau et c'est un registre distribué. Les enregistrements (les utilisateurs) sont partagés avec tous les participants, sans qu'il y'ait besoin d'une autorité centrale et dans lesquels les transactions sont stockées de manière hautement sécurisée vérifiable et permanente, à l'aide de diverses techniques cryptographiques *[5]*.

I.3 Architecture technique de la blockchain

Cette partie se concentre sur l'analyse du cadre technique de la technologie Blockchain.

Après une mise en perspective et une justification de son intérêt pour l'émergence de nouveaux cas d'usage, cette partie décrit les éléments constitutifs de la technologie.

Nous allons présenter les produits à différentes étapes de la blockchain avant de conclure par un panorama de technologies et outils.

Dans la figure I-1, la division de base est faite pour 9 dimensions. Ensuite, nous analysons en détail le contenu spécifique de chaque module *[8]*.

[3] **Transaction :** est tout type d'action impliquée dans la conduite des affaires ou une interaction entre des personnes. Un accord commercial important peut être appelé une transaction, en particulier l'achat ou la vente de marchandises, mais vous pouvez appeler une transaction tout échange avec une autre personne.

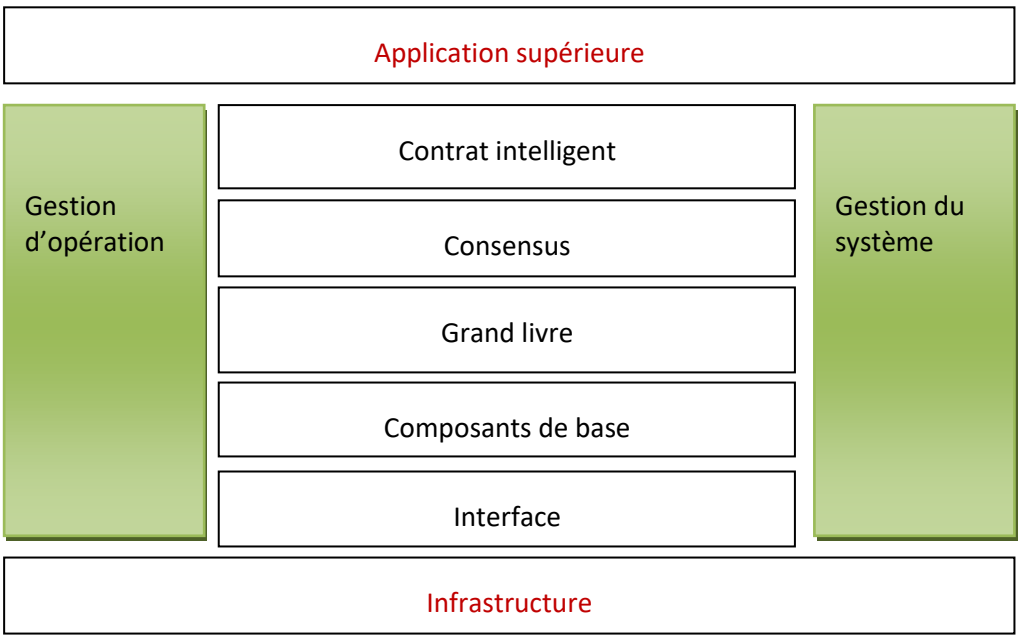


Figure I- 1: Architecture technique de la technologie blockchain

I.3.1 *Infrastructure*

La couche d'infrastructure fournit le système d'exploitation et les installations matérielles y compris les serveurs physiques, les hôtes cloud, etc. pour le fonctionnement normal du système blockchain.
Elle peut être décomposée en trois domaines :

- Ressources informatiques (CPU, GPU, ASIC, etc.).
- Ressources de stockage (disque dur).
- Cette partie des ressources du réseau (bande passante) n'est pas très différente des autres systèmes. Elle n'est rien de plus qu'orientée vers le calcul, certaines orientées vers le stockage et d'autres fortement dépendantes du réseau [9].

I.3.2 *Les composantes de base*

La couche de composants de base implémente l'enregistrement, la vérification et la diffusion d'informations dans le système de blockchain. La blockchain est un système distribué, c'est-à-dire que l'ensemble des blocs signés est répliqué sur tous les nœuds du réseau. Comme tout le monde, vous pouvez, si vous le souhaitez, utiliser votre ordinateur pour en faire un nœud. Pour cela, vous devez télécharger l'ensemble des blocs signés jusqu'à présent. Pour le bitcoin, cela représente déjà plus de 180 giga-octets.

Quand un nœud arrive à signer un nouveau bloc en premier, ce dernier est ajouté à la blockchain de tous les autres nœuds du réseau de manière à toujours avoir une blockchain à jour partout sur le réseau. Il y a énormément de nœuds dans le monde et ils ont tous une copie complète de la blockchain.

- **Découverte du réseau** Le système de blockchain est composé de nombreux nœuds connectés via le réseau, en particulier dans le système de chaîne commun, le nombre de nœuds est souvent important. Chaque nœud doit découvrir les nœuds voisins via le protocole de découverte du réseau et établir une liaison avec les nœuds voisins.
- Une fois que le nœud **émetteur-récepteur de données** est connecté au nœud voisin via le protocole de communication réseau, le module émetteur-récepteur de données termine l'échange de données avec d'autres nœuds. La diffusion des transactions, le consensus des messages et la synchronisation des données sont tous effectués par ce module.
- Les **algorithmes cryptographiques** sont utilisés pour garantir l'intégrité de la blockchain, l'identité du participant, l'authenticité des transactions et parfois la confidentialité du contenu, y compris divers algorithmes de codages, des algorithmes de hachage, des algorithmes de signature, des algorithmes de protection de la confidentialité, etc.
- **Stockage de données** Les données du système de blockchain utilisent différents modes de stockage de données.

Les modèles de stockage incluent des bases de données relationnelles telles que MySQL[3]et des bases de données non relationnelles telles que LevelDB[4].
Généralement, les données à sauvegarder comprennent des données publiques (par exemple : données de transaction, données d'état, etc.) et des données privées locales.

- **Notification de message :** Le module de notification de message fournit des services de notification de message entre différents composants de la blockchain et entre différents nœuds.

Une fois la transaction réussie, le client doit généralement suivre les enregistrements pendant l'exécution de la transaction et obtenir les résultats de l'exécution de la transaction. Le module de notification des messages peut compléter la génération, la distribution, le stockage et d'autres fonctions des messages pour répondre aux besoins du système de blockchain.

I.3.3 G*rand livre*

La couche du grand livre est responsable du stockage. Tous les participants au réseau ont accès au grand livre distribué et à son enregistrement immuable des transactions. Avec ce grand livre partagé, les transactions ne sont enregistrées qu'une seule fois.

La couche de registre incorpore la signature de hachage du bloc précédent dans le bloc suivant pour former une structure de données de chaine de blocs.

La couche du grand livre a deux méthodes d'enregistrement des données basée sur les actifs et basée sur les comptes. Dans un modèle basé sur l'actif, l'actif est modélisé en premier, puis la propriété de l'actif est enregistrée, c'est-à-dire que la propriété est un

[3] **MySQL :** est un système de gestion de base de données relationnelle open source. SQL fait référence au Structured Query Language, le langage de requête utilisé

[4] **LevelDB :** est une bibliothèque de stockage de valeurs-clés rapide écrite par Google qui fournit un mappage ordonné des clés de chaîne aux valeurs de chaîne.

champ de l'actif. Dans le modèle basé sur un compte, un compte est établi en tant qu'objet d'actifs et de transactions, et les actifs sont un champ sous le compte.

Le grand livre possède les avantages suivants :

- Modèle de données basé sur le compte : Il est facile à enregistrer et à interroger les informations liées.
- Modèle de données basé sur les actifs : Ce modèle possède une haute simultanéité afin d'obtenir des performances de traitement simultanées élevées et d'interroger les informations liées au compte dans le temps. Plusieurs plateformes de blockchain migrent vers deux modèles de données ce qui conduit au développement d'un modèle mixte.

I.3.4 *Consensus*

La couche consensus est chargée de coordonner et d'assurer la cohérence des enregistrements de données de chaque nœud dans l'ensemble du réseau c'est-à-dire tous les nœuds sont en compétition pour ajouter le prochain bloc à la blockchain mais un seul d'entre eux sera sélectionné pour le faire (et lui seul sera rémunéré). Cette sélection étant aléatoire pour chaque nouveau bloc. Ce caractère aléatoire est très important pour la sécurité de la blockchain ; puisque personne ne sait quel mineur sera choisi. Il existe différentes façons de réaliser cette sélection aléatoire de mineurs : ce sont les mécanismes de consensus.

Le premier type de mécanisme de consensus est la preuve de travail (POW) utilisée habituellement dans les blockchains publiques de sort que plus la puissance de calcul d'un mineur est élevée, plus la probabilité sélectionnée est grande. Comme la puissance de calcul est coûteuse, le coût d'acquisition de 51% de la puissance totale de calcul du réseau est élevé ; c'est une façon de sécuriser le réseau en rendant le coût d'une attaque disproportionnée par rapport au bénéfice.

Le deuxième type de mécanisme de consensus est utilisé dans les blockchains privées dans lesquelles les nœuds appartiennent à une même entité, ou dans les blockchains semi-privées dans lesquelles les nœuds appartiennent à un consortium de différents utilisateurs autorisés. Ce deuxième type n'a pas besoin d'un mécanisme de consensus aussi coûteux que le POW, car les participants sont connus et de confiance dans une certaine mesure. Dans ce cas, le mécanisme de consensus utilisé est beaucoup plus simple comme l'algorithme de PBFT[5] [9].

Ci-dessous, nous listons la comparaison de quelques algorithmes de consensus [10].

[5] **PBFT :** est un excellent algorithme de consensus pour les consortiums d'entreprises où les membres sont partiellement fiables.

Mécanisme du consensus	Description	Avantages	Inconvénients
POW	Dans une blockchain publique, les ordinateurs des mineurs sont mis à disposition pour résoudre un problème mathématique compliqué. Le 1er qui trouve une solution gagne la récompense du prochain bloc de la chaine.	Simple à mettre en œuvre. Sécurisé. Faible. Consommation des ressources réseau.	Consomme trop de ressources informatiques. La probabilité de bifurcation est élevée. Le consensus prend plus de temps.
POS	Preuve d'enjeu. Les validateurs de transactions doivent mettre en gage la possession de crypto monnaie pour recevoir une récompense. Si un nœud est malveillant, il peut perdre sa mise en gage au profit des validateurs honnêtes	Moins de consommation de ressources.	La mise en œuvre est plus compliquée. Faille de sécurité. Pression de trafic réseau élevée.
BPFT	Consensus dont la liste des validateurs est connue au départ et peut tolérer jusqu'à 1/3 de nœuds compromis (déconnectés ou malveillants).	Consensus de groupe rapide et performant. Pas de fork ou de réorganisation de chaine.	Chaine privée uniquement.
POA	Preuve d'autorité. Consensus dont la liste des validateurs est connue au départ et qui valide à tour de rôle un bloc. Ce type de consensus peut tolérer jusqu'à 49% de nœuds malveillants ou déconnectés.	Consensus de groupe rapide.	Chaine privée uniquement. Fork ou réorganisation de la chaine possible.

Tableau I- 1: Comparaison des mécanismes de consensus de la blockchain

I.3.5 Contrat intelligent

La couche de contrat intelligent est chargée d'implémenter, de compiler et de déployer la logique métier du système blockchain sous forme de code, de terminer le déclenchement des conditions et l'exécution automatique des règles établies et de minimiser l'intervention manuelle.

Les objets opérationnels des contrats intelligents sont principalement des actifs numériques. Les caractéristiques des conditions difficiles à modifier et fortes de déclenchement après la détermination des données déterminent que l'application des contrats intelligents a une valeur élevée et un risque élevé. Comment éviter les risques et exercer une valeur est une difficulté dans l'application à grande échelle actuelle des contrats intelligents.

À l'heure actuelle, l'application des contrats intelligents en est encore à un stade relativement précoce, et les contrats intelligents sont devenus la "zone la plus touchée" pour la sécurité de la blockchain. Du point de vue du temps de sécurité provoqué par les précédentes vulnérabilités des contrats intelligents, il existe de nombreuses failles de sécurité dans le contrat portable, ce qui pose un énorme défi à sa sécurité [11].

À l'heure actuelle, il existe plusieurs idées pour améliorer la sécurité des contrats intelligents :

- La première est la vérification formelle, qui utilise une preuve mathématique stricte pour s'assurer que la logique exprimée par le code du contrat répond à l'intention. Cette loi est stricte en logique, mais difficile, et requiert généralement une agence professionnelle tierce pour effectuer des audits.
- Le second est le chiffrement intelligent des contrats. Les contrats intelligents ne peuvent pas être lus par des tiers en texte brut, ce qui réduit les attaques de contrats intelligents en raison de failles de sécurité logiques. Cette méthode est moins coûteuse, mais elle ne peut pas être open source.
- La troisième consiste à réglementer strictement le format grammatical de la langue du contrat. Résumez l'excellent modèle de contrats intelligents, développez des modèles de contrats intelligents standards et standardisez la préparation des contrats intelligents avec une certaine norme pour améliorer la qualité des contrats intelligents et améliorer la sécurité des contrats.

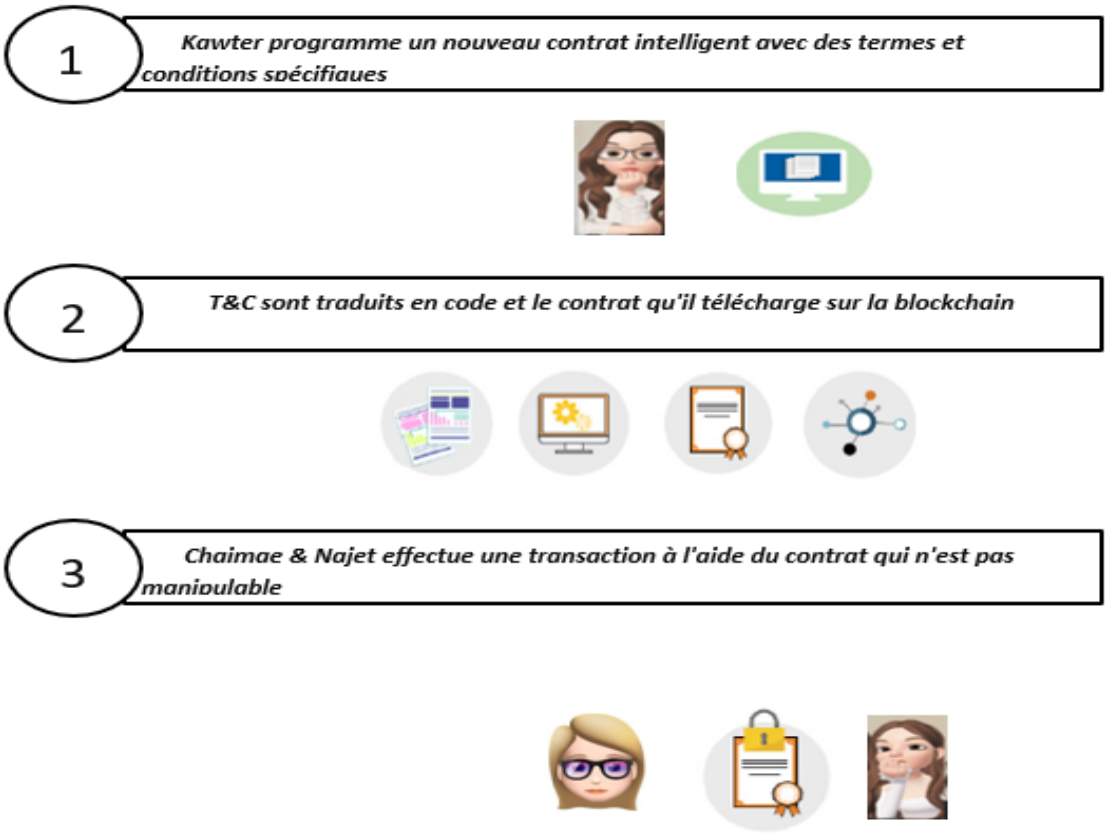


Figure I- 2: Fonctionnement de contrat intelligent

I.3.6 *Gestion de système*

La couche de gestion du système est chargée de gérer d'autres parties de l'architecture de la blockchain, comprenant principalement deux types de fonctions : la gestion des autorisations et la gestion des nœuds.

La gestion des droits est un élément clé de la technologie de la chaîne de blocs, en particulier pour les chaînes d'autorisation qui ont plus d'exigences d'accès aux données.

I.3.7 *Interface*

La couche d'interface est principalement utilisée pour compléter l'encapsulation des modules fonctionnels et fournir une méthode d'appel simple pour la couche d'application.

I.3.8 *Application*

La couche application est la partie qui est finalement présentée à l'utilisateur. Son rôle principal est d'appeler l'interface de la couche contrat intelligent pour s'adapter aux différents scénarios d'application de la blockchain et fournir aux utilisateurs différents services et applications.

I.3.9 *Fonctionnement et maintenance*

L'équipe de gestion de l'exploitation et de la maintenance est principalement responsable de l'exploitation et de la maintenance quotidienne du système blockchain, y compris la journalisation, la surveillance, la gestion et l'expansion.

Sous l'architecture unifiée, les plates-formes traditionnelles ont différents modules de stockage, modèles de données, structures de données, langages de programmation, environnements sandbox[6], etc., en fonction de leurs propres besoins et de leur positionnement.

I.4 Domaine d'application de la Blockchain

La technologie Blockchain, qui réduit les frictions dans les activités commerciales et augmente l'efficacité, révolutionne de nombreuses industries. Et les réformes sont, en même temps, mises en œuvre à grande échelle, impliquant des participants collaborateurs. Dans divers domaines, tels que la finance, les soins de santé et le gouvernement, la blockchain contribue déjà aux réformes de l'industrie. Voici quelques-unes des possibilités infinies de la blockchain.

- **Internet des objets :**
  - *Fret :* Transportez du fret avec plusieurs compagnies maritimes tout en garantissant la transparence et une livraison rapide.
  - *Suivi et conformité des composants :* Stockez les enregistrements sources des pièces d'origine et de rechange dans l'entretien du véhicule.
  - *Consigner les données d'exploitation et de maintenance :* stocker les enregistrements d'exploitation et de maintenance pour les partager avec des partenaires commerciaux ou à des fins réglementaires.
- ***Gestion des identifiants :***
  - Établir une identification numérique de confiance.
- ***Chaîne d'approvisionnement :***
  - Améliorez la traçabilité, la transparence et l'efficacité au sein des réseaux de sécurité alimentaire.
- ***Services financiers :***
  - *Comprendre vos clients :* Accédez à des informations fiables et à jour sur vos clients pour aider les institutions financières à fournir un service client plus précis.
  - *Compensation et règlement :* Accélérez le règlement en transférant des fonds entre les institutions financières directement en temps réel.
  - *Autres exemples :* Lettres de crédit, dette et obligations de sociétés, plateformes de négociation, envois de fonds, accords de mise en pension, devises.

[6] **Sandbox :** est un environnement de test isolé qui permet aux utilisateurs d'exécuter des programmes ou d'exécuter des fichiers sans affecter l'application, le système ou la plateforme sur lesquels ils s'exécutent.

- ***Soins de santé :***
  - Dossiers médicaux électroniques.
  - Banque de virus.
  - Contrat de service et d'assurance REP médecin / vendeur.
  - Blockchain Health Research Commons.
  - Notaire de santé Blockchain.
- ***Assurances :***
  - Traitement des réclamations.
  - Assurance P2P.
  - Propriété.
  - Ventes et souscription.
- ***Agence gouvernementale :***
  - Procédure d'appel d'offres du gouvernement.
  - Votez.
  - Taxe.
- ***Jeux***
- ***La musique :***
  - Streaming plus juste et plus transparent où les musiciens pourraient être payés directement par ceux qui écoutent leur musique.
- ***Les contrats intelligents***
  - L'exécution automatisée d'un contrat ou d'une transaction selon des règles définies

I.5 Pourquoi utiliser la technologie Blockchain

La deuxième question que les gens posent habituellement lorsqu'ils entendent parler de la blockchain est : pourquoi utiliser la blockchain ? Pourquoi utiliser un grand livre distribué ? Pourquoi ne pas utiliser une base de données régulière ou un système hérité comme système d'enregistrement dans ce monde déjà numérique ?

Dans cette partie, nous examinons ce qu'est réellement une blockchain, ce qu'elle peut faire et, surtout, pourquoi utiliser la blockchain ?

I.5.1 *QU'EST-CE QUE LA BLOCKCHAIN ? UN GRAND LIVRE DISTRIBUÉ*

Juste au cas où vous auriez besoin d'un petit rattrapage, les gens parlent souvent de « blockchain » au singulier, comme s'il n'en était qu'un. En réalité, ils devraient parler de la technologie de la chaîne de blocs également connue sous le nom de technologie de registre distribué ou DLT ou des chaînes de blocs au pluriel, car il en existe de nombreuses différentes, y compris les chaînes de blocs publiques (sans autorisation) et privées (avec autorisation).

La blockchain Bitcoin, la blockchain Ethereum, la blockchain NEO[7] et bien d'autres sont des exemples de blockchains publiques et de technologie de blockchain distribuée. Les

[7] **NEO :** est une plate- forme d'application décentralisée blockchain open source fondée en 2014 par Da HongFei et Erik Zhang. Depuis son changement de nom à NEO d'Antshares en 2017, la vision du projet est de

chaînes de blocs autorisées sont adaptées à une utilisation d'entreprise ou organisationnelle, avec un tel exemple étant la chaîne de blocs Hyperledger[8] d'IBM[9]. Mais nous y reviendrons dans un instant.

La blockchain est un moyen simple mais ingénieux de transmettre des informations de A à B de manière entièrement automatisée et sûre. Une partie à une transaction lance le processus en créant un bloc. Ce bloc est vérifié par des milliers, voire des millions d'ordinateurs répartis sur le Net. Chaque bloc de la chaîne est chronologiquement connecté aux blocs précédents et synchronisé avec les nœuds du réseau, créant non seulement un enregistrement unique, mais la falsification d'un seul block signifierait la falsification de la chaîne de block entière ce qui rend très difficile la falsification [13].

I.5.2 *Comment fonctionne la Blockchain*

Les tâches que nous effectuons sur les appareils numériques peuvent être divisées en deux catégories : transactionnelles et non transactionnelles. Écrire des e-mails, regarder des vidéos et naviguer sur Internet sont principalement des activités non transactionnelles, ce qui signifie que nous n'avons rien acheté ou vendu, ni signé aucun accord contractuel.

Cependant, nous effectuons de plus en plus de transactions en ligne, telles que la signature des contrats, l'achat d'article.

Les transactions numériques sont plus rapides et plus pratiques, mais peuvent ne pas être sécurisées, permettant aux cybercriminels de se connecter à notre compte ou d'obtenir nos numéros de sécurité sociale et d'autres informations sensibles. La blockchain est conçue pour agir comme un compte public virtuel qui peut être consulté par tout le monde et écrit à l'encre persistante. Chaque bloc est **un fichier** et un nouveau bloc est créé toutes les 10 minutes, qui contient un enregistrement de toutes les transactions précédentes, répertoriées dans l'ordre et se terminant par une nouvelle transaction [14].

En terme technique, la blockchain est une base de données distribuée basée sur des Merkle-Trees chiffrés, c'est-à-dire que la base de données n'est ni créée, ni développée, ni stockée dans une unité centrale de traitement. Au lieu de cela, il existe une copie de chaque ordinateur ou "nœud" utilisé pour traiter et vérifier les transactions. Lorsqu'une nouvelle transaction est ajoutée, toutes les copies sont modifiées en même temps. Si une transaction qui ne respecte pas les règles de protocole est détectée par les nœuds du réseau, elle est immédiatement expulsée [15].

---

réaliser une « économie intelligente » en utilisant la technologie de la blockchain et des contrats intelligents pour émettre et gérer des actifs numérisés.

[8] **Hyperledger d'IBM :** Hyperledger Fabric est le cadre de la plate-forme IBM Blockchain. Pour permettre une innovation rapide et une adoption plus rapide dans tous les secteurs, la plate-forme IBM Blockchain est basée sur un protocole de chaîne de blocs open source.

[9] **IBM** International Business Machines : une société multinationale d'informatique et de technologie de l'information

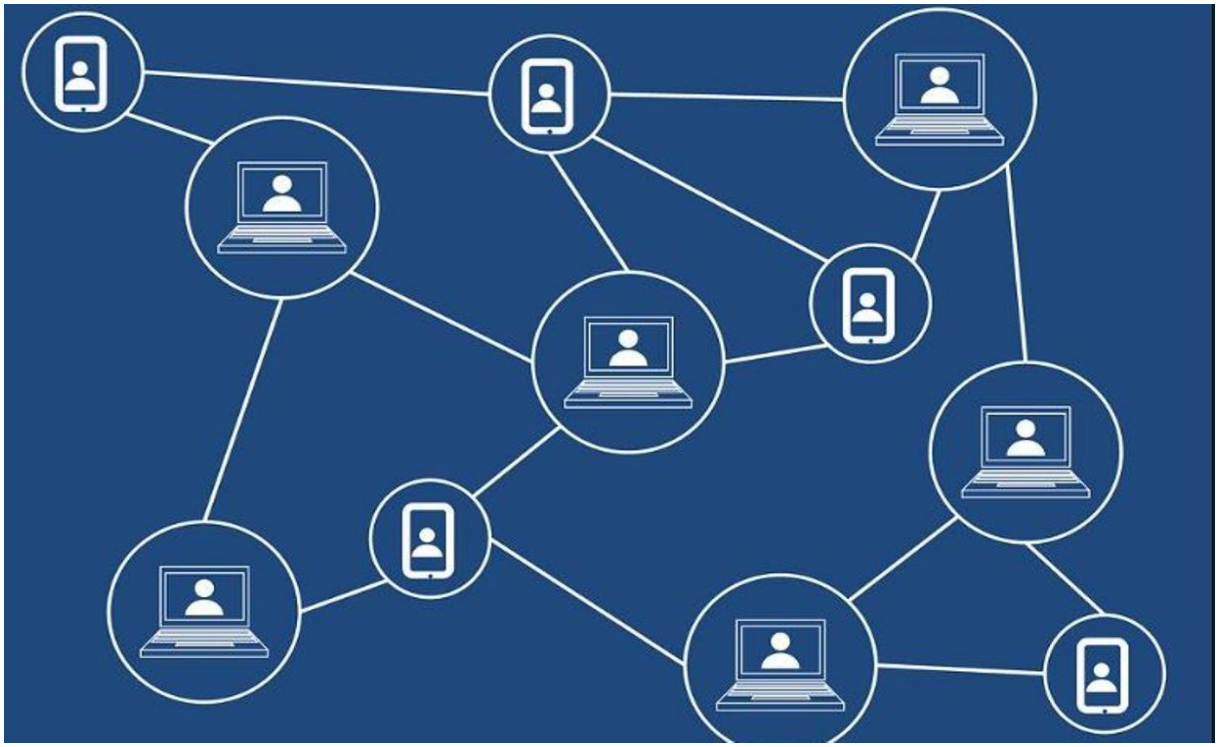


Figure I- 3: Base de données de la blockchain

La blockchain la plus connue est celle du réseau Bitcoin. Lorsqu'une transaction est générée via un certain nœud, la transaction doit être transmise à d'autres nœuds pour la vérification et puis confirmée par des mineurs. L'action ne peut être annulée. Ce procédé consiste à chiffrer les données de transaction via des signatures numériques et à obtenir une série de valeurs de hachage uniques représentant la transaction via la fonction de hachage, puis à diffuser cette valeur de hachage à d'autres nœuds participants dans le réseau de blockchain Bitcoin pour la vérification (voir figure ci-dessous). Chaque nœud effectue le calcul de preuve de travail (POW) pour déterminer qui peut vérifier la transaction. Le nœud qui a obtenu le droit de vérification diffuse le bloc à tous les nœuds qui terminent le POW dès que possible et diffuse son propre bloc aux autres nœuds. A ce stade, d'autres nœuds confirmeront si les transactions contenues dans ce bloc sont valides. Après avoir confirmé qu'elles n'ont pas été dépensées à plusieurs reprises et ont des signatures numériques valides, elles acceptent le bloc. En ce moment, le bloc est officiellement connecté à la chaîne de blocs [16].

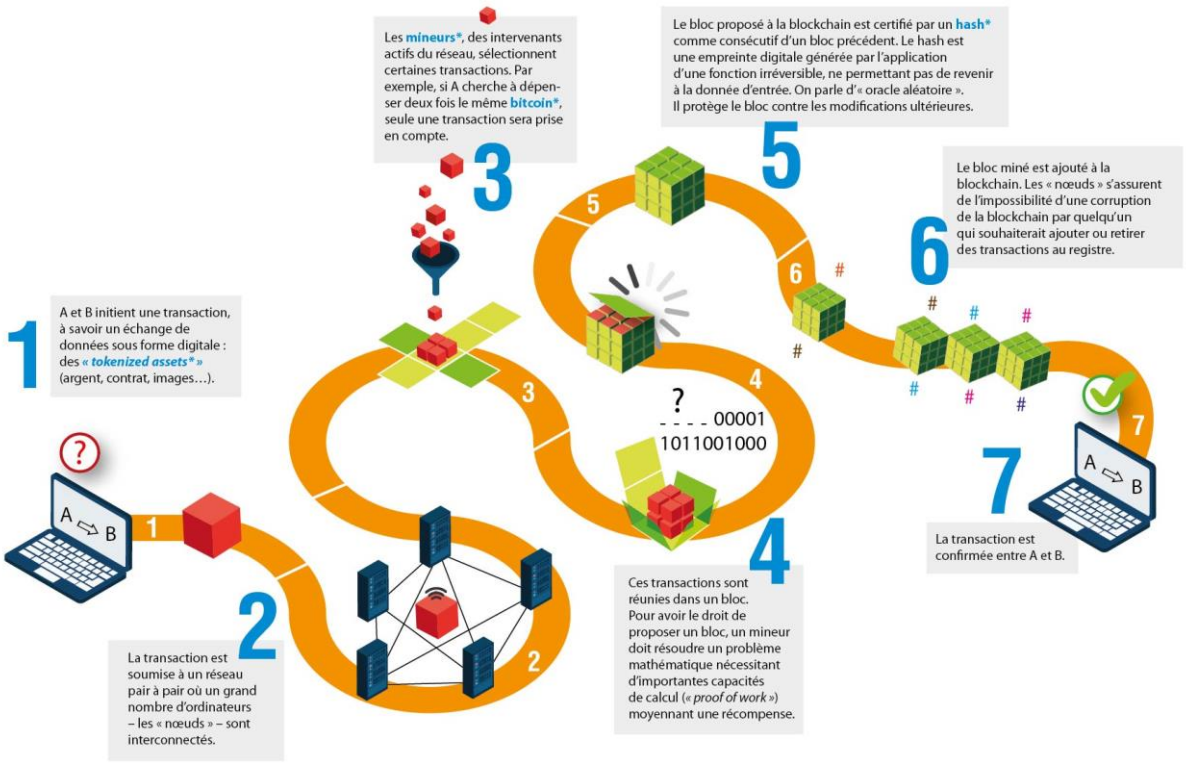


Figure I- 4: Fonctionnement de la Blockchain

I.5.3 *Blockchain vs Base de données normales*

On dit que la blockchain est une sorte de base de données, donc « Quelle est la différence entre la blockchain et la base de données traditionnelle ? »

I.5.3.1 Centralisation vs Décentralisation système

La raison même pour laquelle nous examinons le débat entre centralisation et décentralisation est que la blockchain est conçue pour être décentralisée. Cependant, les termes décentralisée et centralisée ne sont pas toujours clairs. Donc, la plupart des concepts et exemples de cette section sont inspirés des notes de M. Vilatik Buterin, le fondateur de la blockchain Ethereum.

Qu'est-ce qu'un système distribué alors ? Un système distribué centralisé est un système dans lequel il y a, par exemple, un nœud maître chargé de décomposer les tâches ou les données et de répartir la charge entre les nœuds. D'autre part, un système distribué décentralisé est un système où il n'y a pas de "maître" [16]. Donc, la blockchain en est un exemple, et nous en examinerons de nombreuses représentations schématiques plus loin dans ce chapitre.

La figure suivante représente un schéma d'un système distribué centralisé.

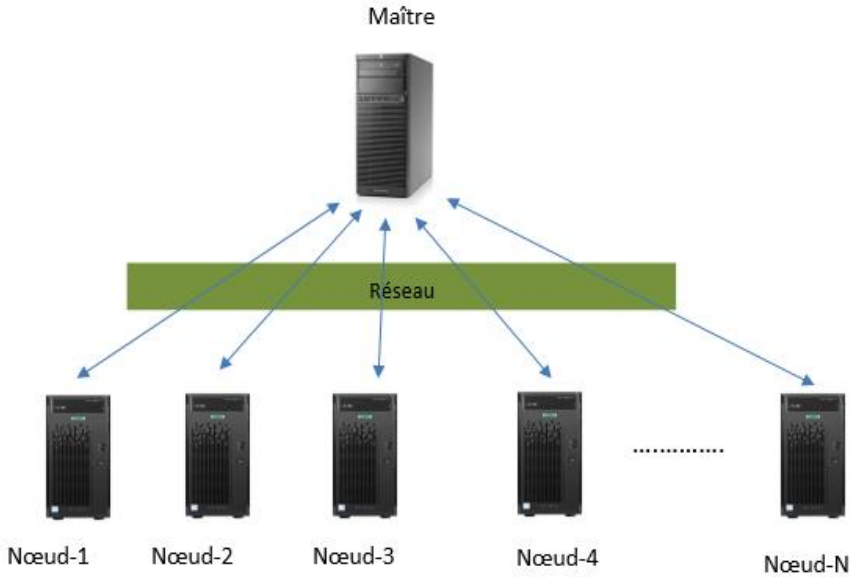


Figure I- 5: Système distribué avec contrôle centralisé

Cette représentation est similaire à la mise en œuvre de Hadoop[10], à titre d'exemple. Si le calcul est plus rapide dans de telles conceptions en raison de l'informatique distribuée, il souffre également de limitations dues à la centralisation.

Il est extrêmement important de noter qu'un système centralisé / décentralisé ne se limite pas à l'architecture technique.

Ce que nous entendons dire, c'est qu'un système peut être décentralisé techniquement, mais ne peut-être pas aussi logiquement ou politiquement. Jetons un œil à ces différentes perspectives pour pouvoir concevoir correctement un système en fonction de l'exigence :

Architecture Technique : un système peut être centralisé ou décentralisé à partir d'une architecture technique c'est-à-dire combien nous considérons le nombre de nœuds (ordinateurs) utilisés pour concevoir un système.

Perspective politique : elle indique le contrôle qu'un individu ou un groupe de personnes ou bien une organisation dans son ensemble sur un système. Si les nœuds du système sont contrôlés par eux, donc le système est centralisé.

[10] **Hadoop :** est un cadre logiciel open source pour le stockage de données et l'exécution d'applications sur des clusters de matériel de base. Il offre un stockage massif pour tout type de données.

Perspective logique : un système peut être logiquement centralisé ou décentralisé en fonction de son apparence, qu'il soit centralisé ou décentralisé techniquement ou politiquement.

- Le réseau de distribution de contenu, d'autre part, est décentralisé sur le plan architectural, également décentralisé sur le plan logique, mais est un système politiquement centralisé car il appartient à des entreprises. On cite à titre d'exemple Amazon et CloudFront[11].
- Examinons maintenant la blockchain. Les objectifs de la blockchain étaient de permettre la décentralisation. Ainsi, il est décentralisé architecturalement par conception. Il est également décentralisé d'un point de vue politique, car personne ne le contrôle.

Une petite mise au point technique, parce qu'on voit qu'il y a beaucoup de confusion sur les termes « centralisé », « décentralisé », « distribué », « pair à pair », pour pouvoir comprendre pourquoi la blockchain est décentralisée par conception [17].

Commençons par décrire le système centralisé.

I.5.3.1.1 Système centralisé

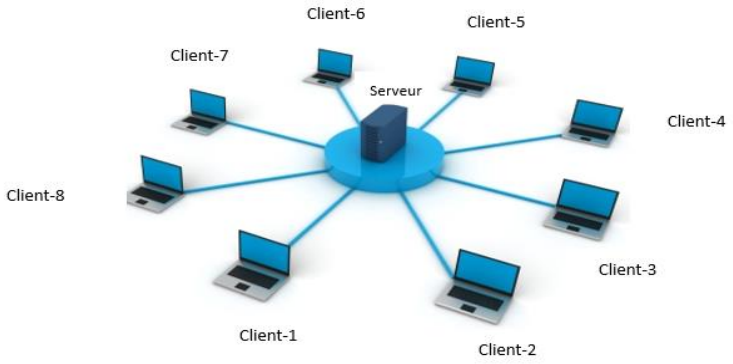


Figure I- 6: Système centralisé

C'est un système avec un contrôle centralisé avec tous les autorités administratives, un serveur a priori dans le cas informatique [18]. Un système centralisé typique peut apparaître comme le montre la figure ci-dessous.

Comme le montre la figure tous les nœuds de l'application sont hébergés sur une seule machine et les utilisateurs peuvent directement se connecter à la machine centrale.

Le principal problème du système centralisé est qu'il n'est pas facilement évolutif. Il y a une limite au nombre de CPU dans un système et finalement le système entier doit être mis à niveau ou remplacé.

[11] **CloudFront :** est un réseau de diffusion de contenu proposé par Amazon Web Services. Les réseaux de distribution de contenu fournissent un réseau mondialement distribué de serveurs proxy qui mettent en cache le contenu, comme les vidéos Web ou d'autres médias volumineux, plus localement pour les consommateurs, améliorant ainsi la vitesse d'accès pour télécharger le contenu

I.5.3.1.2 Système décentralisé

Comme l'indique son nom, ce système n'a pas de centre. L'idée pour un système de communication, c'est que tous les nœuds puissent être une partie d'un réseau qui n'a pas d'autorité principale, et que ces autorités puissent parler entre elles [19].

Un système décentralisé typique peut apparaître comme le montre la figure suivante :

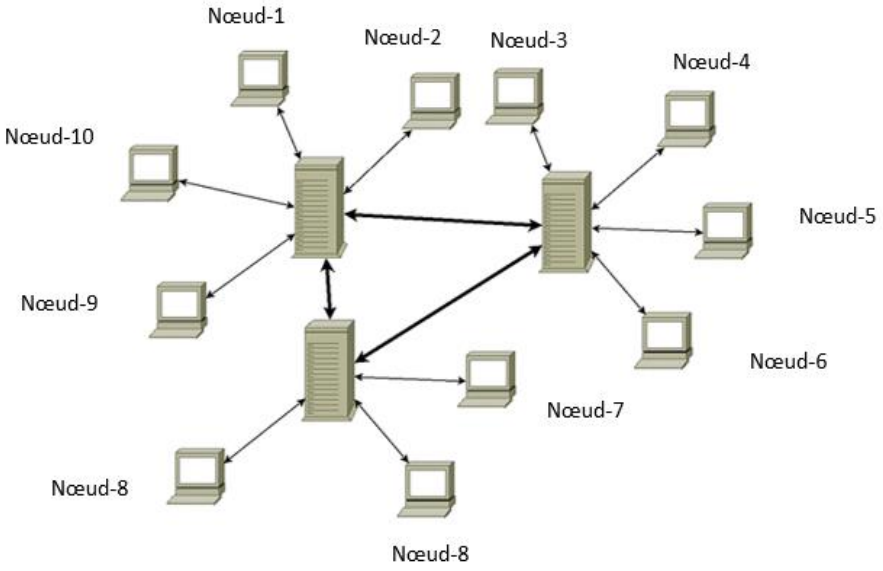


Figure I- 7: Système décentralisé

On note qu'un système distribué peut également être décentralisé. Par exemple, la blockchain ! Cependant, contrairement aux systèmes distribués communs, la tâche n'est pas subdivisée aux nœuds, car aucun maître ne le ferait dans la blockchain. Pour le réseau P2P ou « pair à pair », l'idée de cette architecture est de permettre au réseau de fonctionner même si on lui coupe l'accès à une partie de lui-même. Un système peer-to-peer est illustré par la figure I-8.

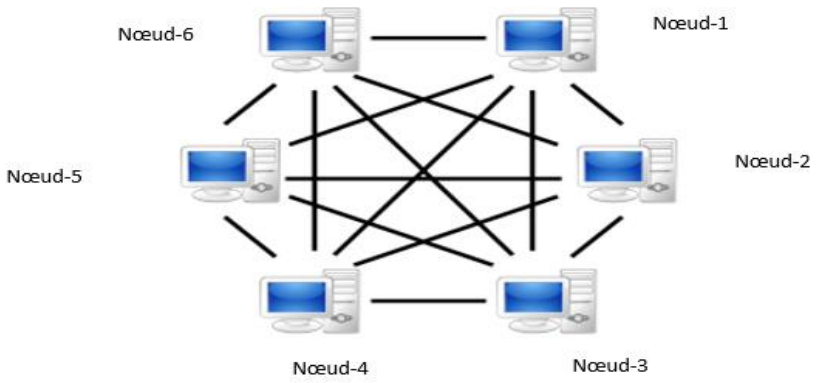


Figure I- 8: Système décentralisé avec p2p architecture

Ce qui fait principalement la différence entre un système « décentralisé » et « entièrement P2P » c'est la place des serveurs. Pour transformer votre système décentralisé avec serveurs en un système entièrement P2P, vous mettez un seul client sur votre serveur, et vous mettez le serveur et le client sur la même machine [20].

I.6 BLOCKCHAINS AUTORISÉS VS BLOCKCHAINS PUBLIQUES

Vous ne pouvez pas être un crypto investisseur ou entrepreneur sans avoir une vraie compréhension des différences entre les types de blockchains ainsi que leurs implications. Donc, Quelle sont les différences entre blockchain privée et blockchain publique ? et Comment faire le choix d'une ou de l'autre de ces options ?

Une blockchain est dite publique c'est à dire « ouverte » lorsque n'importe qui peut devenir membre du réseau. La chaîne d'alliance est ouverte à des organisations spécifiques. De nos jours, l'industrie estime généralement que la chaîne d'alliance se situe entre les chaînes publiques et privées et appartient essentiellement à la catégorie des chaînes privées.

Nous nous concentrerons sur les chaînes publiques et privées, car les chaînes d'alliance peuvent être classées comme chaînes privées [21].

I.6.1 Blockchain publique

La chaîne publique est une chaîne de blocs qui peut être lue par n'importe qui dans le monde, n'importe quelle personne peut envoyer des transactions et les transactions peuvent être effectivement confirmées, et n'importe qui peut participer au processus de consensus. Le processus de consensus détermine quels blocs peuvent être ajoutés à la blockchain [22]. En effet, les transactions qui sont stockées dans ces « Ledgers » sont non modifiables, non supprimables, accessibles en lecture par tout le monde, et ultra sécurisées à base de la cryptographie.

I.6.1.1 Les caractéristiques de la blockchain publique

- Protéger les utilisateurs des développeurs.
- Faible barrière à l'accès.
- Toutes les données sont publiques par défaut.

I.6.2 Blockchains privées

Les blockchains privées regroupent toutes les solutions que les entreprises peuvent installer dans leur contexte interne et qu'elles vont décider d'utiliser de manière partiellement décentralisée, c'est-à-dire elle fait référence à une blockchain dont les autorisations d'écriture sont entre les mains d'une seule organisation [23].

I.6.2.1 Les caractéristiques de la blockchain privée

- La vitesse de transaction est très rapide.
- Meilleure protection de la vie privée.
- Les coûts de transaction sont considérablement réduits voire nuls.
- Aide à protéger ses produits de base contre les dommages.

I.7 Mécanismes de sécurité dans les réseaux

I.7.1 *Introduction*

De nos jours, toutes les informations sont numérisées et il est possible de s'y référer en utilisant un ordinateur personnel à portée de main. L'utilisation généralisée d'Internet a permis d'obtenir les informations nécessaires à tout moment et en tout lieu. Mais en même temps, les problèmes de sécurité augmentent. À l'heure actuelle, un réseau privé virtuel (VPN), qui peut accéder en toute sécurité à un réseau interne de l'extérieur de l'entreprise ou construire un réseau virtuel et sécurisé entre les succursales, s'est généralisé. Cependant, il existe de nombreuses méthodes VPN, chacune avec des caractéristiques différentes.

I.7.2 *Qu'est-ce qu'un VPN (Virtual Private Network)*

Le VPN est une technologie permettant de construire virtuellement un réseau privé en utilisant la technologie d'authentification, la technologie de cryptage, la technologie de tunneling, etc. sur un réseau public partagé par de nombreuses personnes comme la ligne Internet [24].

I.7.3 *Les protocoles du VPN*

I.7.3.1 OpenVPN

Il s'agit d'un protocole open source relativement nouveau, fiable et exemplaire.

Très populaire parmi les services tiers, il n'y a pas de support natif pour la plateforme.

Il prend en charge une variété d'algorithmes, garantissant le plus haut niveau de sécurité.

C'est l'un des protocoles les plus rapides. La vitesse de communication dépend du niveau de cryptage, mais dans la plupart des cas, l'effet n'est pas ressenti en utilisation normale.

Bien qu'il puisse sembler difficile à configurer, de nombreux excellents services VPN sont conçus pour nécessiter peu ou pas d'intervention de l'utilisateur [25].

I.7.3.2 IKEv2 (Internet Key Exchange v2)

IKEv2 est un protocole de tunneling basé sur IPsec développé par Microsoft et Cisco. Il est stable et sécurisé car il prend en charge la capacité de reconnexion et divers algorithmes.

Il possède une très bonne vitesse de communication relativement plus rapide que L2TP, SSTP, PPTP.

Il n'est pas souvent disponible sur d'autres plates-formes, à l'exception des terminaux BlackBerry. Parce qu'il s'agit d'une technologie propriétaire, certaines personnes peuvent vouloir l'utiliser en fonction de leur connaissance de Microsoft. Mais la même version open source existe [26].

I.7.3.3 L2TP (protocole de tunneling de couche 2)

Il vient du Cisco L2F et du PPTP de Microsoft. Puisqu'il n'y a aucune fonction de sécurité, elle est généralement fournie avec IPsec.

Les terminaux et les systèmes d'exploitation qui prennent en charge les VPN récents sont toujours intégrés. Dans l'ensemble, c'est un bon protocole, mais il a été souligné que la NSA (National Security Agency) pourrait interférer avec une fuite récente. Il n'y a pas beaucoup d'avantages par rapport à OpenVPN [27].

I.7.3.4 SSTP (Secure Socket Tunneling Protocol)

SSTP est introduit pour la première fois par Microsoft sur Windows Vista. Il est intégré à Windows et peut ne pas être disponible sur d'autres plateformes. La plupart des pares-feux peuvent être facilement traversés. En raison de la technologie Microsoft, vous ne pouvez pas dire où vos données sont envoyées [28].

La vitesse de communication est rapide et relativement sécurisée, mais les vulnérabilités qui la sous-tendent la rendent inférieure au meilleur protocole.

I.7.3.5 PPTP (protocole de tunneling point à point)

Il s'agit du premier protocole VPN pris en charge par Windows. Il peut être utilisé sur tous les appareils pouvant utiliser un VPN.

La vitesse de communication est très élevée en raison des faibles normes de cryptage.

PPTP offre une très faible sécurité. On sait qu'il est facilement gêné par la NSA depuis de nombreuses années. Microsoft a amélioré PPTP, mais recommande d'utiliser d'autres protocoles, tels que SSTP et L2TP / IPSec.

I.7.3.6 Comparaison entre les protocoles [29]

Protocol	La vitesse	Cryptage & Navigation sécurisée	La stabilité	Streaming multimédia	Partage de fichiers P2P	Compatible avec	Avantages	Inconvénients
PPTP	Vite	Faible	Moyen	Bien	Bien	La plupart des OS et appareils.	Haute vitesse.	Ancien protocole. Faible sécurité.
L2TP/IPSec	Vite	Moyen	Bien	Bien	Bien	La plupart des OS et appareils.	La sécurité est forte.	Vitesse lente. Peut être bloqué par un pare-feu
IKEv2/IPSec	Vite	Bien	Bien	Bien	Bien	La plupart des OS et appareils.	La sécurité est forte. Haute vitesse. Suivez automatiquement la communication	Moins d'appareils pris en charge.
OpenVPN	Vite	Bien	Moyen	Bien	Bien	La plupart des OS et appareils	La sécurité est forte. Haute vitesse Open source. Peut passer à travers le pare-feu	Peut nécessiter une application tierce.
SSTP	Moyen	Bien	Moyen	Moyen	Bien	Windows	La sécurité est bien. Non bloqué par le pare-feu.	Est opaque Pour Windows

Tableau I- 2: Comparaison entre les protocoles du VPN

I.8 Conclusion

Dans ce chapitre, nous avons couvert l'évolution de la blockchain, son histoire et sa définition, quels sont les avantages de la conception et son fonctionnement, pourquoi est-il si important avec certains cas d'utilisation pertinents et cette dernière est divisé en deux partie privée et publique et nous conclurons par les mécanismes de sécurité dans les réseaux.

Dans le chapitre **II**, nous approfondirons dans les fondamentaux de la blockchain.

Chapitre II: Fonctionnement de la blockchain

Partie 1 : Les crypto-monnaies

II.1 Introduction

Avant de commencer, on fait une petite définition sur laquelle on parle de la crypto currency, ou bien le chiffrement de l'argent [30] (en français : crypto-monnaie, également traduit en **monnaie de cryptographie**, **monnaie de mot de passe**) qui utilise des principes de **la cryptographie** pour assurer la sécurité des transactions et des unités de contrôle des transactions pour créer **un moyen d'échange** [31].

Tout comme les billets de banque généralement utilisés dans la conception anti-contrefaçon, l'anti-contrefaçon de la crypto-monnaie est un nouveau type de jeton qui utilise la **crypto-monnaie** et le **hachage numérique** à l'aide de **la monnaie numérique** et de **la monnaie virtuelle** et est lié à des contrats intelligents. En 2009, **Bitcoin** est devenu la première monnaie de cryptage **décentralisée**, multidevises.

Après, ce terme fait référence à chiffrer ces conceptions [32].Depuis lors, plusieurs crypto-monnaies similaires ont été créées, souvent appelées **altcoins**[12] [33][34][35].Les crypto-monnaies sont basées sur un mécanisme de consensus décentralisé [36][37], contrairement aux systèmes financiers bancaires qui reposent sur un système de réglementation centralisé. Vous pariez avant tout sur un projet, une promesse, voire un modèle de société. Pour vous aider à y voir plus clair, nous avons décidé de vous expliquer les différences qui existent entre les diverses familles de crypto-monnaies en fonction de leur utilisation la plus logique [38].

II.2 Quelles différences entre Bitcoin (BTC), Litecoin (LTC) et Ethereum (ETH) ?

Le Bitcoin[13], le Litecoin[14] et l'Ethereum[15] sont les trois principales monnaies cryptographiques. Cependant, si elles ont de nombreux points communs, elles disposent

[12] Le terme « **altcoins** » fait référence à toutes les crypto-monnaies autres que Bitcoin.

[13] Le **Bitcoin** (₿, BTC) (de l'anglais bit : unité d'information binaire et coin « pièce de monnaie ») est une cryptomonnaie autrement appelée monnaie cryptographique.

[15] **Litecoin** (symbole monétaire : Ł ; sigle : LTC) est une monnaie électronique distribuée, Chaque Litecoin est divisé en cent millions d'unités plus petites, définies par huit décimales.

[16] **Ethereum** : est un protocole d'échanges décentralisés permettant la création par les utilisateurs de contrats intelligents grâce à un langage Turing-complet.

chacune de leurs propres particularités. Nous allons donc voir quelles sont les différences entre ces trois crypto-monnaies [39].

II.2.1 *Les différences entre le Bitcoin et le Litecoin*

Ce sont les deux monnaies qui se ressemblent le plus. Rien d'étonnant à cela, Litecoin était l'une des premières fourchettes Bitcoin. Il a été développé par Charlie Lee et a été initialement lancé en tant que code open source sur GitHub[16] le 7 octobre 2011. Le jeton est conçu comme une alternative plus rapide et évolutive au Bitcoin. Ce jeton est très similaire au jeton précédent, mais il existe de nombreuses différences importantes. Étant donné que le jeton a été conçu comme une alternative "plus légère" au Bitcoin, il a été modifié pour améliorer sa vitesse et sa disponibilité [40].

Certaines différences entre Litecoin et Bitcoin sont :

- Temps de génération de blocs réduit (de 10 minutes pour Bitcoin à 2,5 minutes pour Litecoin)
- Augmentation de l'offre totale de jetons
- Différents algorithmes de hachage (Scrypt au lieu du Sha-256 de Bitcoin)
- Interface utilisateur graphique améliorée
- Le litecoin est souvent appelé la crypto-monnaie "argentée" pour correspondre au statut "or" de Bitcoin.

Le tableau II-1 montre les principales différences entre Bitcoin et Litecoin [40].

	Bitcoin	**Litecoin**
Restrictions de jeton	21 Millions	84 Millions
Algorithme	SHA-256	Scrypt
Concept	Digital money (Gold)	Digital money (Silver)
Temps de sortie moyen	10 min	2.5 min
Explorateur de blocs	Blockchain.info	Block-explorer.com
Qui a réussi	Satoshi Nakamoto	Charlie Lee
Date de création	3 janvier 2009	7 octobre 2009

Tableau II- 1: La comparaison entre Bitcoin & Litecoin

[16] **GitHub** : est une société mondiale basée aux États-Unis qui propose un hébergement pour le contrôle des versions de développement logiciel à l'aide de Git.

II.2.2 *Les différences entre le bitcoin et l'Ethereum*

L'Ethereum et le Bitcoin sont deux crypto-monnaies avec plusieurs points communs mais surtout de grosses différences. Voici donc les principales d'entre elles [41]:

- Entre le Bitcoin et l'Ethereum, la principale différence est la durée que prend la génération d'un bloc. Alors que le Bitcoin prend 10 minutes pour le faire, l'Ethereum est bien plus rapide puisqu'il lui faut seulement 12 secondes.
- L'algorithme utilisé par les deux crypto-monnaies est différent. De plus, l'Ethereum est beaucoup plus souple et laisse un plus grand nombre d'options à ses utilisateurs. En résumé, en utilisant l'Ethereum, on peut faire tout ce qu'on peut faire avec le Bitcoin, mais on peut également faire d'autres choses.
- Enfin, l'Ethereum est beaucoup plus orienté vers sa technologie, dont l'Ether n'est qu'une composante alors que le Bitcoin a surtout créé une monnaie stable qui est avant tout destinée à faire de l'échange.

II.2.3 *Les différences entre le Litecoin et l'Ethereum*

En ce qui concerne le Litecoin et l'Ethereum, on peut noter que les différences sont similaires à celles entre l'Ethereum et le Bitcoin. Cependant, on pourra remarquer quelques différences importantes entre les deux [42] :

- Entre le Litecoin et l'Ethereum, la principale différence est la durée que prend la génération d'un bloc. Il faut moins de temps pour générer un bloc avec l'Ethereum qu'avec le Litecoin.
- Si le Litecoin était auparavant l'une des crypto-monnaies les plus rentables pour les mineurs, elle est actuellement dépassée par l'Ethereum.

En somme, on peut dire que :

- Chacune de ces trois crypto-monnaies dispose de ses propres avantages et particularités.
- Cependant, le Bitcoin est de loin la crypto-monnaie la plus prisée, malgré la polyvalence de l'Ethereum et l'attractivité du Litecoin. Il le doit particulièrement à sa très grande stabilité.
- Parmi les différentes crypto-monnaies, Bitcoin sert de référence et les autres lui sont comparées.
- Vous trouverez différents types de crypto-monnaies, qui peuvent vous permettre de payer de manière anonyme, de miner, de trouver une bonne alternative à Bitcoin ou qui proposent leur propre écosystème.

II.2.5 Comment acheter des crypto-monnaies ?

Lorsqu'on veut commencer à utiliser des crypto-monnaies, il peut parfois être difficile de savoir comment s'y prendre pour effectuer son premier achat. Nous allons donc vous guider pas à pas et vous expliquer comment y parvenir [43].

Donc, La « crypto » dans les crypto-monnaies fait référence à une **cryptographie** compliquée qui permet la **création** et le **traitement des monnaies** numériques et de leurs **transactions** à travers des systèmes décentralisés. Parallèlement à cette importante caractéristique, la cryptographie sert à déterminer la capacité à sécuriser les transactions sur

internet. Il est désormais possible d'opérer, dans un système ouvert, un réseau totalement sécurisé et totalement décentralisé de transactions anonymes entre un très grand nombre de participants qui ne se connaissent pas et ne se font pas mutuellement confiance. Ces progrès ouvrent des perspectives nombreuses pour le stockage et la transmission confidentielle de données sur internet. Il est assez naturel que des entrepreneurs innovants s'appuient sur ces technologies pour développer des expériences monétaires et financières [44].

A l'heure actuelle ou tous les pays s'interrogent aujourd'hui sur la meilleure manière de répondre aux défis posés par les crypto-monnaies [44], l'Algérie se doit tracer une voie originale, préservant les bénéficies de l'innovation technologique et protégeant l'intégrité des marchés. Pour y parvenir, il faut répondre aux trois questions soulevées par la crypto-monnaies : Que sont-elles ? Par la cryptographie et comment la blockchain fonctionne ?

II.3 Qu'est-ce qu'une Crypto-monnaie :

La crypto-monnaie est un terme général pour les devises qui utilisent la cryptographie comme mesure de sécurité, synonyme de monnaie virtuelle [45].

Elle se caractérise par l'utilisation de technologies telles que le cryptage à clé publique, le hachage ou les signatures numériques incorporant les deux en tant que technologies pour améliorer la confidentialité des communications [46].

Une monnaie virtuelle peut être utilisée comme prix pour des biens ou des services entre un nombre indéterminé de personnes et d'entreprises sur Internet. Dans la loi sur le règlement des fonds, les crypto-monnaies sont juridiquement définies comme suit [47] :

- Lors de l'achat ou de l'emprunt de biens ou de la réception de services, les biens peuvent être utilisés par des personnes non spécifiées pour payer ces frais, et les personnes non spécifiées sont achetées en tant qu'autre partie. Et la valeur de la propriété qui peut être vendue.
- Valeur du bien qui peut être échangée avec une personne non spécifiée en tant qu'autre partie avec celles énumérées au point précédent et qui peut être transférée à l'aide d'une organisation de traitement électronique de l'information.

« De nombreuses crypto-monnaies sont des systèmes décentralisés basés sur la technologie blockchain, distribués par un réseau d'ordinateurs disparates. Elle n'est émise par aucune autorité centrale, ce qui le rend théoriquement à l'abri de l'ingérence ou de la manipulation du gouvernement. La première crypto-monnaie basée sur une chaîne de blocs était le Bitcoin, qui reste la crypto-monnaie la plus populaire et la plus précieuse [48]. »

II.3.1 La double innovation des cryptomonnaies ?

Hormis les pièces et billets de banque[17], si vous avez suivi des activités bancaires, d'investissement ou de crypto-monnaie au cours des dix dernières années, vous connaissez peut-être la « blockchain », la technologie de tenue de registres derrière le réseau Bitcoin. Et il y a de fortes chances que cela ait autant de sens [49]. En essayant d'en savoir plus sur la

[17] Qui représentent généralement moins de 10% de la masse monétaire totale (10% dans la zone euro)

blockchain, vblockchain, vous avez probablement rencontré une définition comme celle-ci : "la blockchain est un grand livre public distribué et décentralisé".

La révolution est ailleurs plus profonde, à la fois technologique et monétaire. Ces deux éléments sont présents dans la plupart des cryptomonnaies.

II.3.1.1 L'innovation technologique :

Plusieurs technologies de base sont utilisées par les crypto-monnaies. La technologie blockchain est une technologie qui permet de structurer les données, d'identifier et de suivre les transactions, de partager ses informations sur un réseau distribué d'ordinateurs, créant ainsi un « réseau de confiance distribué » [50]. Le fondement des crypto-monnaies comme le Bitcoin, la Blockchain fournit un moyen transparent et sécurisé pour suivre la propriété et le transfert des actifs.

- *La technologie des registres distribués (DLT) :* Groupes de personnes et d'entreprises qui partagent les mêmes valeurs, s'associent pour créer une plateforme centrée sur la communauté. En fonction des contributions, chaque membre est récompensé [50].
- *La blockchain est une forme de technologie de registre distribué*, dans laquelle les données sont regroupées en chaines de blocs successifs. Lorsque nous prononçons les mots « bloc » et « chaîne » dans ce contexte, nous parlons en fait d'informations numériques (le « bloc ») stockées dans une base de données publique (la « chaîne »). L'ensemble des blocs vise à cristalliser le contenue des transactions. Le bloc est positionné dans la blockchain de manière à rendre impossible[18] toute modification accidentable. Cette cristallisation repose sur un lien cryptographique, et établie entre chaque bloc et le suivant. Cette architecture venant à souder tous ces éléments entre eux [51].
- *Les procédures de consensus :* elles permettent de définir la manière dont les nœuds doivent interagir, la façon de transmettre les données entre ces derniers et les exigences pour une validation de bloc réussie.

Ces trois innovations technologiques ne sont pas liées dans tous les crypto-monnaies. On le retrouve intégralement dans le Bitcoin et Ether, mais pas dans d'autres crypto-monnaies. Par Exemple Ripple[19].

Le schéma suivant permet de situer la technologie blockchain dans le cadre existant :

[18] La fonction de Hachage

[19] **Ripple** est une **altcoin** et un protocole de paiement utilisant la technologie blockchain pour faciliter les transactions financières des banques.

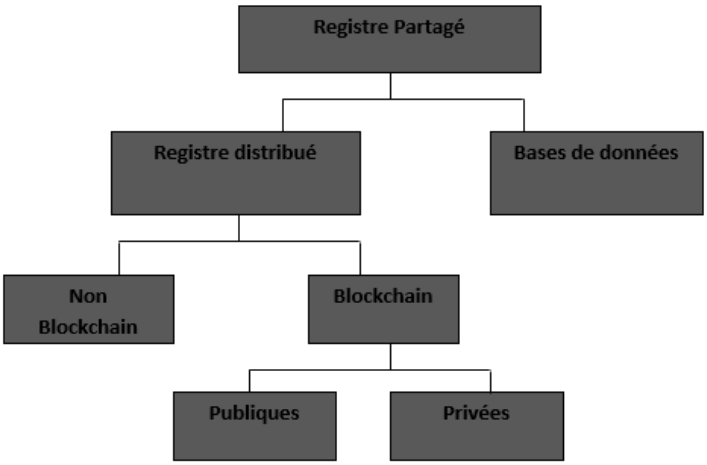


Figure II- 1: Innovation technologique de la technologie blockchain

La digitalisation de la valeur ; est une deuxième innovation que les crypto-monnaies contribuent à promouvoir. Elle permet de transférer en toute sécurité entre individus sans aucun intermédiaire à l'aide de la digitalisation « billet de banque ». Cette représentation digitale appelée des « jetons » ou bien par leur nom anglais, des « Tokens » [52].

II.3.1.2 L'innovation monétaire

Pour mieux comprendre, il faut décrire les caractéristiques d'un système monétaire actuel et de mesurer en quoi les crypto-monnaies se distinguent.

La banque et la monnaie ; Cette monnaie dite « billet de banque » à plusieurs caractéristiques notables :

Pour utiliser la monnaie, vous devez disposer d'un compte bancaire, nécessaire à l'incorporation financière au sein d'une entreprise ;

- Toutes les transactions monétaires suivent un mouvement bancaire. Donc, elles peuvent être surveillées et règlementées ;
- L'argent est une créance pour une personne morale spécifique identifiable, dans ce cas, la banque.
- La monnaie de la banque est une monnaie « privée ». Donc, elle est vulnérable à une perte de confiance.
- Elle bénéficie par le biais de mécanisme d'assurance ou d'un accès au refinancement de la banque centrale ;
- Par ailleurs, la banque centrale émet la monnaie servant de « base »au système.

Avec le développement d'internet, et après l'envolée des paiements en ligne, on voit arriver de nouvelles solutions de paiement très modernes [53].

La monnaie électronique est un substitut à **l'argent liquide (pièces et billets),** stocké dans un dispositif électronique, magnétique ou sur un serveur distant.

La monnaie électronique peut également être stockée (et utilisée) sur les téléphones portables. Les informations de paiement sont alors stockées sur la carte SIM du Smartphone et font l'objet d'un cryptage de sécurité et utilisent la technologie NFC (Near Field

Communication) pour autoriser des paiements, ou encore sur un compte de paiement en ligne.

Les crypto-monnaies ; est une monnaie virtuelle qui n'existe que numériquement et s'appuie sur la cryptographie pour sa sécurité. Difficile à contrefaire grâce à cette caractéristique, son principal attrait est sa nature organique. Elle utilise un système décentralisé pour enregistrer les transactions et gérer l'émission de nouvelles unités. Une crypto-monnaie n'est émise par aucune autorité centrale, ce qui la rend théoriquement non vulnérable à l'ingérence ou à la manipulation du gouvernement [53].

Les créateurs des crypto-monnaies ont accordé beaucoup d'importance et d'attention à leur régime d'émission. Ces régimes, assez divers combinent un mélange de rigueur[20], d'ambiguïté[21] et d'innovation.

II.3.2 *La diversité et l'évolution de la crypto-monnaie*

Il y a 11 ans, la première transaction Bitcoin était née. En une courte période de 11 ans, l'industrie des crypto-monnaies s'est développée rapidement et, en même temps, de nouveaux actifs cryptographiques sont apparus et ont également gagné un grand nombre de supporters.

L'industrie a radicalement changé depuis 6 ans, et le Bitcoin n'est plus la seule crypto-monnaie. L'émergence d'un grand nombre de crypto-monnaies nous a également conduits à classer la valeur de marché des crypto-monnaies.

La société d'analyse de données DataLight[22] a réalisé une telle vidéo, retraçant les changements dans la capitalisation boursière de diverses crypto-monnaies depuis 2013.

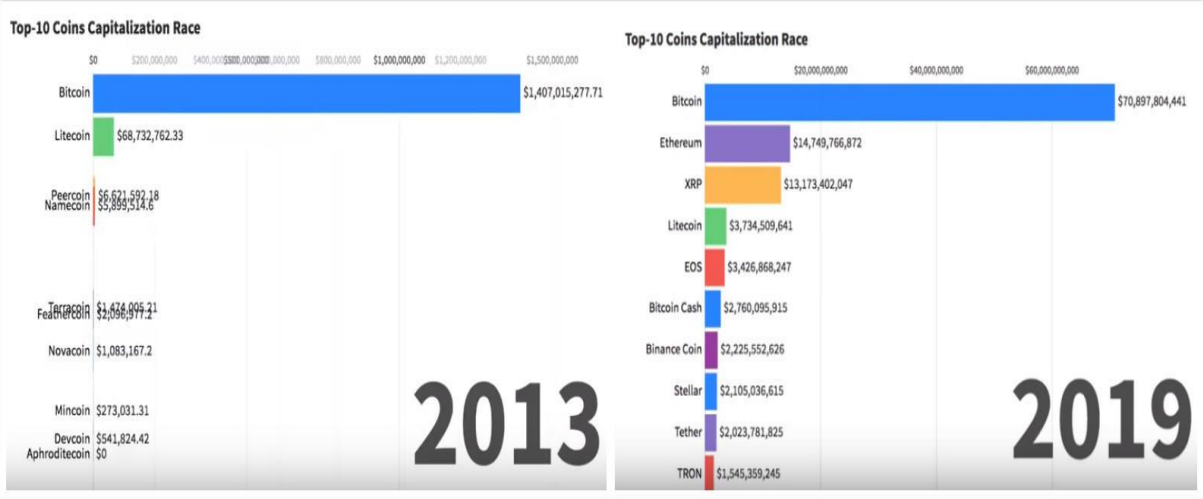


Figure II- 2: Comparaison de la valeur marchande des crypto-monnaies en 2013 et 2019

[20] **La rigueur** provient d'un encadrement de la quantité de monnaie émise.

[21] **L'ambiguïté** vient des conditions dans lesquelles certains fondateurs se « réservent », à l'émission c'est une fraction de stock de crypto-monnaie.

[22] **DataLight** fournit des données uniques sur les crypto- monnaies. Il donne toutes les informations pour prendre des décisions commerciales ultra précises

Les crypto-monnaies sont évolutives et peuvent se transformer ou se multiplier à partir d'une même souche selon le processus des « forks[23] ». La fourche des crypto-monnaies est une branche de la blockchain.

II.3.2.1 Les types des forks

Il existe deux types de fourches : les fourches dures et les fourches souples. Au niveau le plus élémentaire, ces fourches se produisent en raison des méthodes de travail incohérentes des participants sur le protocole spécifique de la blockchain. Cela est généralement dû à un groupe de développeurs souhaitant modifier les règles que la blockchain utilise pour vérifier la validité des transactions. Ce groupe de développeurs peut choisir de changer une partie du protocole, puis la blockchain sera divisée en deux voies possibles.

II.3.2.1.1 Fourche dure (hard fork*)*

Un hard fork[24] se produit lorsqu'une nouvelle règle est introduite dans le protocole et il doit être définitivement séparé de la version actuelle de la blockchain. Par conséquent, il nécessite que tous les nœuds soient mis à niveau vers la dernière version. En bifurquant de cette manière, un chemin suivra la nouvelle blockchain mise à niveau. L'autre façon continue comme avant. Bitcoin Cash (BCH) est un exemple de la première fourche dure de la blockchain Bitcoin.

II.3.2.1.2 Fourche souple (soft fork)

Les fourches souples, en revanche, sont rétro compatibles avec tous les blocs précédents de la blockchain. Cela signifie que les nœuds existants sur le réseau n'ont pas besoin de mettre à niveau ou d'accepter de nouveaux protocoles, et les nouvelles transactions sont valides. Bien que les fourches souples puissent également diviser la chaîne de blocs, la probabilité que cela se produise est beaucoup plus faible. Pour cette raison, il est souvent utilisé pour implémenter des mises à niveau logicielles plus petites.

Partie 2 : La cryptographie

II.4 Définition

La cryptographie est le composant le plus important de la blockchain. C'est un domaine de l'utilisation des mathématiques pour crypter et décrypter des données [54]. La

[23] Ce terme « **Forks** » désigne la scission d'une blockchain donnant naissance à deux nouvelles chaines partageant le même historique.

[24] Un « **Hard Fork** » permet de rompre définitivement avec la précédente version de la blockchain, en déviante la chaîne de blocs initiale sur la base d'un nouveau protocole créé à cet effet.

Le « **soft fork** » s'apparente pour sa part à une simple modification du protocole existant, mais sans création de monnaie nouvelle, le protocole antérieur pouvant être encore utilisé.

cryptographie vous permet de stocker des informations sensibles ou de les transmettre sur des réseaux non sécurisés comme Internet afin qu'elles ne puissent être lues par personne, à l'exception du destinataire prévu [55]. Nous essaierons de développer une solide compréhension de certains des concepts cryptographiques de cette section, car différents problèmes peuvent nécessiter des solutions cryptographiques différentes ; c'est le composant le plus important pour assurer la sécurité du système. De nombreux piratages ont été signalés sur les portefeuilles et les échanges en raison d'une conception plus faible ou de mauvaises implémentations cryptographiques.

La cryptographie existe depuis plus de deux mille ans. Maintenant, c'est la science de garder les choses confidentielles en utilisant des techniques de cryptage. Cependant, la confidentialité n'est pas le seul objectif. En matière de cyber sécurité, il y a un certain nombre de choses qui nous préoccupent en ce qui concerne les données. Ceux-ci incluent la **confidentialité**, **l'intégrité**, la **disponibilité** et la **non-répudiation**, comme mentionné dans la liste suivante, que nous explorerons plus loin :

- *La confidentialité* signifie que nos données ne peuvent pas être consultées / lues par des utilisateurs non autorisés.
- L'intégrité signifie que nos données nous parviennent à 100% intactes et n'ont pas été modifiées, que ce soit par un acteur malveillant, une perte de données ou autre.
- *La disponibilité (authentification)* signifie que nos données sont accessibles en cas de besoin.
- La non-répudiation : l'expéditeur, après avoir envoyé un message, ne peut pas nier ultérieurement qu'il a envoyé le message. Cela signifie qu'une entité (une personne ou un système) ne peut pas refuser la propriété d'un engagement ou d'une action antérieure.

Toute information sous forme de message texte peut être appelée texte en clair. L'idée est de chiffrer le texte en clair à l'aide d'un algorithme de chiffrement et d'une clé qui produit le texte chiffré. Le texte chiffré peut ensuite être transmis au destinataire prévu, qui le déchiffre en utilisant l'algorithme de déchiffrement et la clé pour obtenir le texte en clair [56].

Prenons un exemple, Kawter veut envoyer un message (m) à Chaima, si elle envoie simplement le message tel quel, tout adversaire, disons, peut facilement intercepter le message et la confidentialité est compromise. Donc, Kawter veut crypter le message en utilisant un algorithme de cryptage (E) avec une clé secrète (k) pour produire le message chiffré appelé "texte chiffré". L'ennemie doit connaître l'algorithme (E) et la clé (k) pour intercepter le message. Plus l'algorithme et la clé sont forts, plus l'ennemie a de la difficulté à attaquer. Notez qu'il serait toujours souhaitable de concevoir des systèmes de blockchain leurs systèmes de sécurité s'adapteront également afin de répondre aux besoins des différentes utilisations.

L'ensemble d'étapes commun à cette approche peut être représenté comme illustré à la figure suivante :

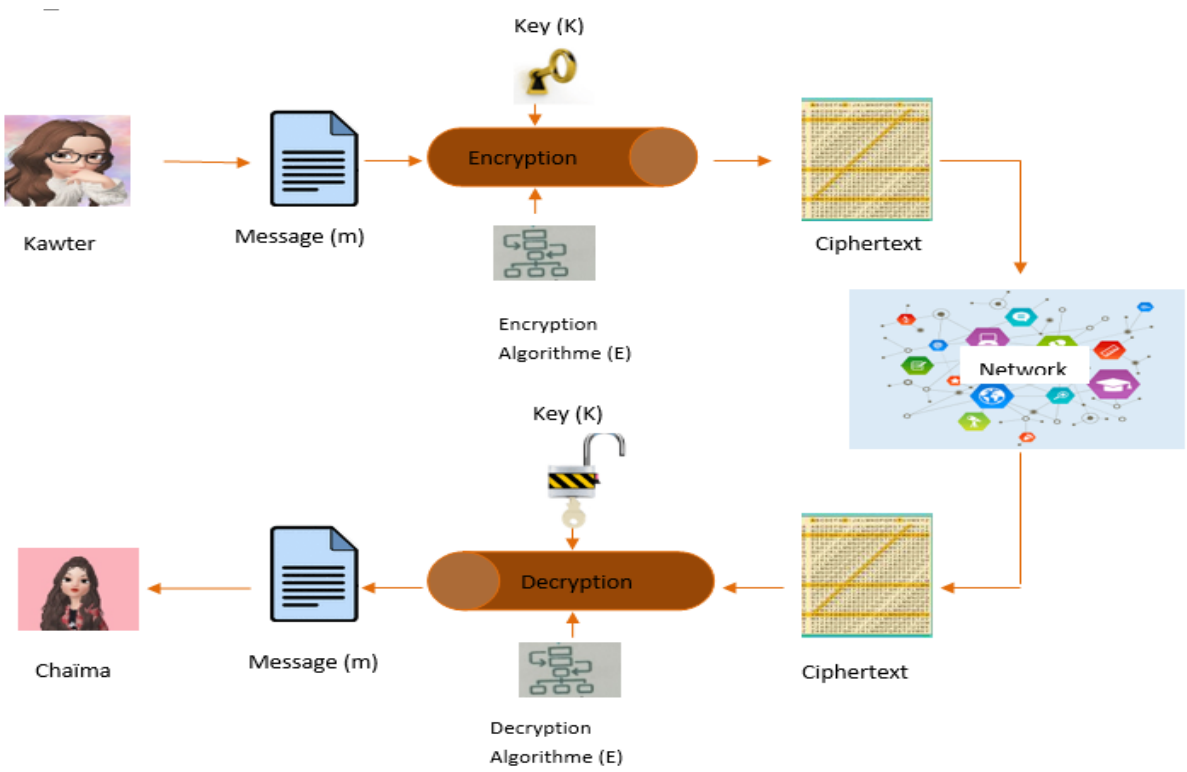


Figure II- 3: Comment fonctionne la cryptographie en général

II.5 Les différents types de la cryptographie

Nous examinerons les différentes formes de cryptographie numérique et com ment elles peuvent nous aider à atteindre les trois autres objectifs énumérés ci-dessus. Lorsque nous parlons de cryptographie numérique, nous nous référons généralement à l'un des éléments suivants :

- Cryptage symétrique.
- Cryptage asymétrique.
- Fonctions de hachage.
- Signatures numériques.

Nous avons développé chacun de ces points ci-dessous. Gardez également à l'esprit que ces exemples sont destinés à illustrer les concepts et non à fournir les meilleures pratiques en matière de sécurité des données.

II.5.1 Cryptographie à clé symétrique

II.5.1.1 Introduction

En cryptage, des clés symétriques (également appelée cryptographie à clé commune, cryptographie à clé secrète ou cryptographie classique) sont utilisées pour crypter et décrypter des informations. En d'autres termes, pour déchiffrer les informations, vous avez besoin de la clé utilisée pour le chiffrement [57]. Une "clé" ici est en fait un secret partagé entre plusieurs parties, ce qui permet de maintenir un lien vers des informations confidentielles. Cette exigence selon laquelle les deux parties ont accès à la clé privée est

l'un des principaux inconvénients du cryptage à clé symétrique par rapport au cryptage à clé publique [57].

Bien que la vitesse de traitement de la cryptographie à clé symétrique soit élevée, à mesure que le nombre de personnes qui échangent des données augmente, le nombre de clés à gérer augmente.

Il existe deux principaux types de cryptage symétrique dans les temps modernes : le cryptage des flux de données et le cryptage des blocs de données. Le cryptage du flux de données consiste à générer un flux de code aléatoire avec l'algorithme et la clé, puis à générer le flux de données crypté avec le flux de données XOR. La partie de déchiffrement n'a besoin que de générer le même flux de code aléatoire. Le chiffrement des blocs de données divise les données d'origine en blocs de données de taille fixe (tels que 64 bits), et le chiffreur utilise la clé pour traiter les blocs de données. Le chiffrement des flux de données est généralement plus rapide, mais le chiffrement par bloc est plus sécurisé. Dans les méthodes de cryptage courantes, DES et 3DES sont les méthodes de cryptage par blocs les plus utilisées, AES est une méthode de cryptage par blocs plus récente, RC4 est le cryptage des flux de données, etc [58].

Le schéma de chiffrement le plus courant utilisant la cryptographie symétrique comprend cinq composants :

- *Texte brut :* informations originales.
- *Algorithme de chiffrement :* prenez la clé comme paramètre, effectuez diverses règles et étapes de substitution et de conversion sur le texte en clair et transformez le résultat en texte chiffré.
- *Clé :* Les paramètres des algorithmes de chiffrement et de déchiffrement, qui affectent directement le résultat de la transformation du texte en clair.
- *Ciphertext :* résultat de la transformation du texte en clair.
- *Algorithme de déchiffrement :* la transformation inverse de l'algorithme de chiffrement, avec le texte chiffré comme entrée et la clé comme paramètres, et le résultat de la conversion est en texte clair.

Il existe plusieurs opérations mathématiques couramment utilisées en cryptographie symétrique. Le but commun de ces opérations est de brouiller les chiffres en clair du texte en clair aussi profondément que possible, augmentant ainsi la difficulté de déchiffrement. Parmi ces opérateurs, on a **XOR** connu sous le nom de "**exclusive or**", il s'agit d'une opération d'algèbre booléenne binaire. On peut également comprendre simplement que si les deux chiffres participant à l'opération OU exclusif sont égaux, le résultat est 0 et s'ils ne sont pas égaux, le résultat est 1. Le symbole mathématique de XOR est $\oplus$ [59].

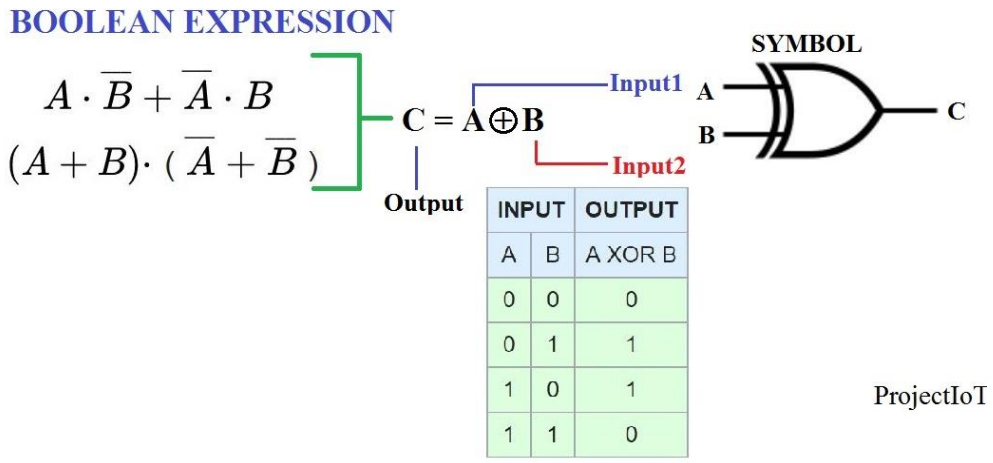


Figure II- 4: Opérateur XOR et sa table de vérité

II.5.1.2 Les algorithmes de chiffrement

Les algorithmes de chiffrement célèbres incluent "DES", "RC4" et "AES". Dans ce qui suit, on va parler seulement des algorithmes "DES" et "AES".

II.5.1.2.1 Data Encryption Standard « DES »

DES est un chiffrement par blocs typique - un algorithme qui convertit un texte en clair de longueur fixe en un texte chiffré de la même longueur à travers une série d'opérations complexes. Pour DES, la longueur de bloc est de 64 bits. Dans le même temps, DES utilise des clés pour personnaliser le processus de transformation, de sorte que l'algorithme estime que seuls les utilisateurs qui détiennent la clé utilisée pour le chiffrement peuvent déchiffrer le texte chiffré. La clé est de 64 bits en surface, cependant, seuls 56 d'entre eux sont réellement utilisés pour l'algorithme, et les 8 bits restants peuvent être utilisés pour la parité et rejetés dans l'algorithme. Par conséquent, la longueur de clé effective de DES n'est que de 56 bits [60].

Deux principes de conception de chiffrement par blocs sont utilisés dans la conception du DES : la confusion et la diffusion, dont le but est de résister à l'analyse statistique du crypto système par l'adversaire. Le rôle de la diffusion est d'appliquer l'influence de chaque texte en clair à plus de bits de texte chiffré en sortie le plus rapidement possible, afin d'éliminer la structure statistique du texte en clair dans un grand nombre de textes chiffrés et de rendre l'influence de chaque clé aussi rapide que possible. Il y a plus de bits de texte chiffré pour éviter que la clé ne soit décodée pièce par pièce.

Le cadre algorithmique du chiffrement DES est le suivant :

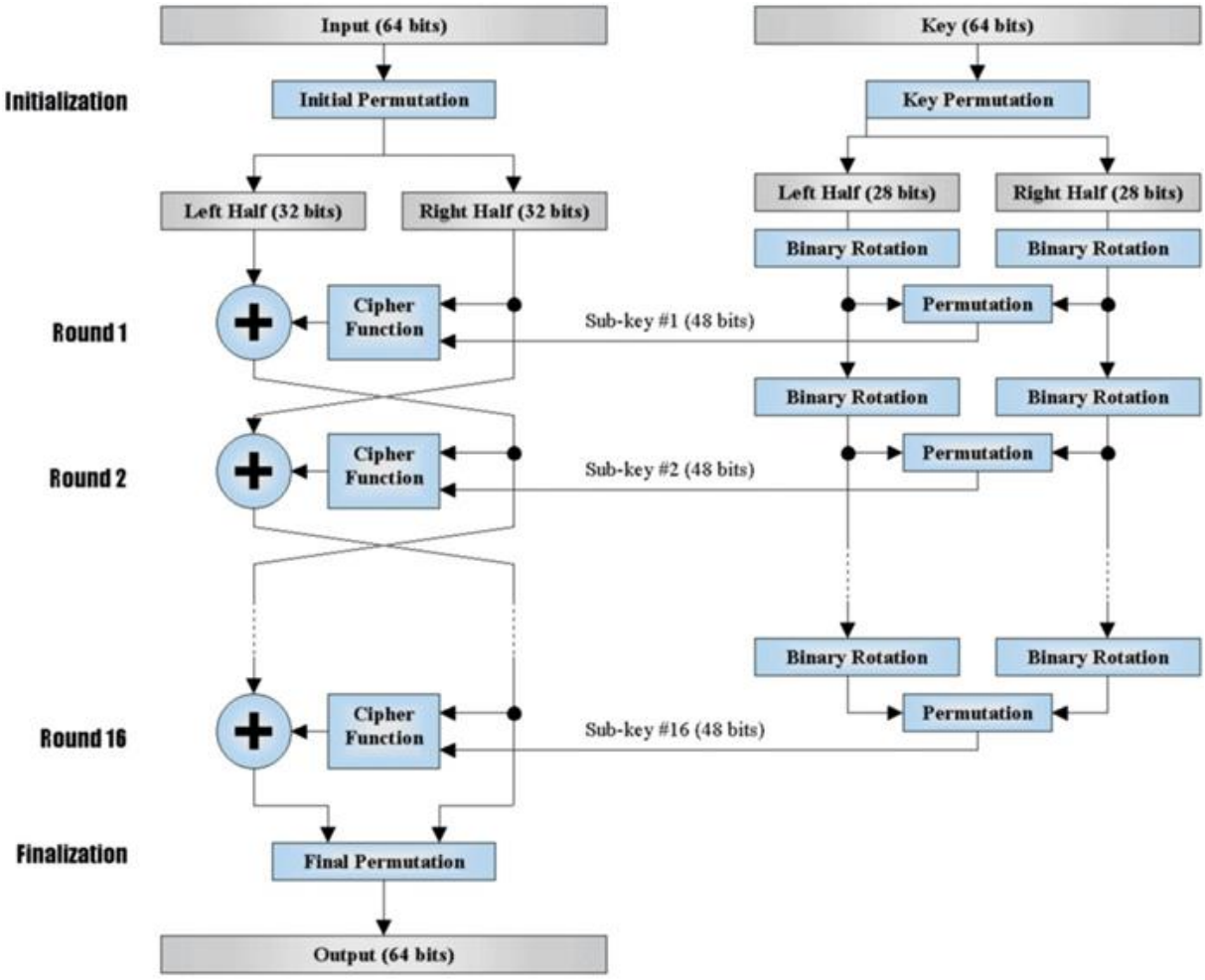


Figure II- 5: Structure DES

La vitesse des recherches exhaustives de clés contre DES après 1990 a commencé à gêner les utilisateurs de DES. Cependant, les utilisateurs ne voulaient pas remplacer DES car il faut énormément de temps et d'argent pour changer les algorithmes de chiffrement qui sont largement adoptés et intégrés dans de grandes architectures de sécurité. L'approche pragmatique n'était pas d'abandonner complètement le DES, mais de changer la manière dont le DES est utilisé. Cela a conduit aux schémas modifiés de Triple DES.

TDES (ou 3DES) est un algorithme qui améliore DES. Il utilise trois clés de 64bits pour chiffrer trois fois les données, et sa sécurité est améliorée par rapport à l'algorithme DES. Cependant, les performances de l'algorithme TDES sont bien inférieures à celles de l'algorithme AES, et son efficacité d'implémentation logicielle et matérielle n'est pas aussi bonne que l'algorithme AES. En 1999, le National Institute of Standards and Technologie (NIST) a désigné TDES comme algorithme cryptographique pour la période de transition. A l'heure actuelle, l'utilisation des algorithmes TDES diminue, et on prévoit qu'ils seront éventuellement remplacés par des algorithmes AES.

II.5.1.2.2 Advanced Encryption Standard « AES »

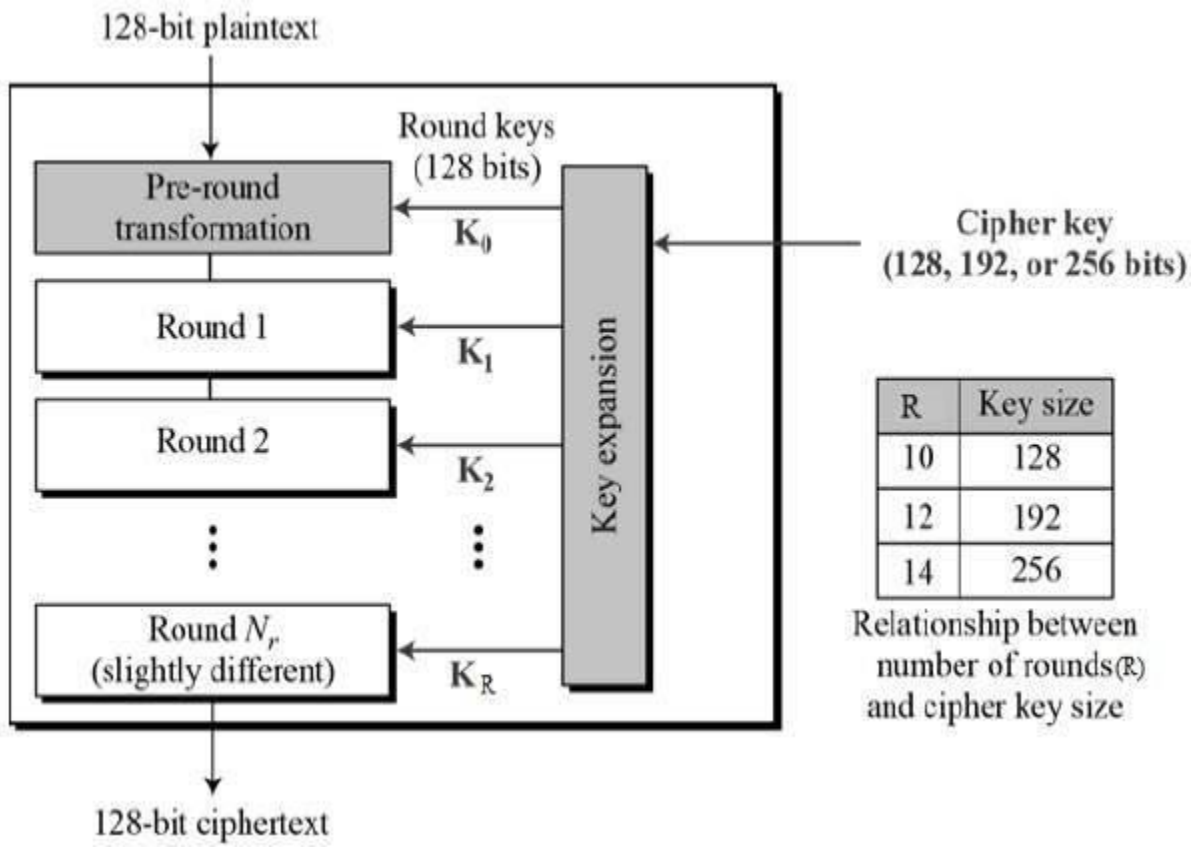
L'Institut national des normes et de la technologie (NIST) a travaillé à l'élaboration d'une nouvelle norme de chiffrement pour sécuriser les informations gouvernementales. L'organisation en est aux dernières étapes d'un processus ouvert de sélection d'un ou

plusieurs algorithmes ou formules de brouillage des données pour la nouvelle norme de chiffrement avancé (AES) et prévoit de prendre des décisions.

AES est destiné à être un successeur plus fort et plus efficace de la norme de chiffrement des données triple (3DES), qui a remplacé le DES vieillissant, qui a été fissuré en moins de trois jours en juillet 1998.

AES est une nouvelle génération de normes de chiffrement NIST / FIPS[25]. Au cours des 30 prochaines années, il remplacera DES en tant que norme de cryptage de mot de passe commune pour les entreprises publiques et privées. Le texte chiffré crypté peut protéger 100 ans de sécurité.

Il est utilisé pour protéger les données sensibles. Le bloc de données est de 128 bits, et la longueur de la clé peut être sélectionnée parmi 128/192/256 bits.



R	Key size
10	128
12	192
14	256

Figure II- 6: Structure AES

Le processus de cryptage AES fonctionne sur une matrice 4 × 4 octets. Cette matrice est également appelée « état » et sa valeur initiale est un bloc de texte en clair. La taille d'un élément de la matrice est l'un des blocs de texte en clair. La méthode de chiffrement Rijndael prend en charge des blocs plus grands et le "numéro de ligne" de sa matrice peut

[25] **FIPS (Federal Information Processing Standards**) sont des normes annoncées publiquement développées par le National Institute of Standards and Technology pour être utilisées dans les systèmes informatiques par des agences gouvernementales américaines non militaires et des entrepreneurs gouvernementaux.

être augmenté selon les besoins. Lors du chiffrement, chaque tour de boucle de chiffrement AES (sauf le dernier tour) comprend 4 étapes [61] :

- *AddRoundKey :* chaque octet de la matrice est associé à un sous-octet retour alliage de clé fait (clé année) de l'opérateur OU EXCLUSIF, chaque sous-clé générée par le système de génération de clé.
- *SubBytes :* Grâce à une fonction de remplacement non linéaire, chaque octet est remplacé par un octet correspondant par une table de recherche.
- *ShiftRows :* tournez chaque ligne de la matrice.
- *MixColumns :* opérations pour mélanger complètement les lignes individuelles dans la matrice. Cette étape utilise une transformation linéaire pour mélanger les quatre octets de chaque ligne. L'étape MixColumns est omise dans la dernière boucle de chiffrement et remplacée par une autre AddRoundKey.

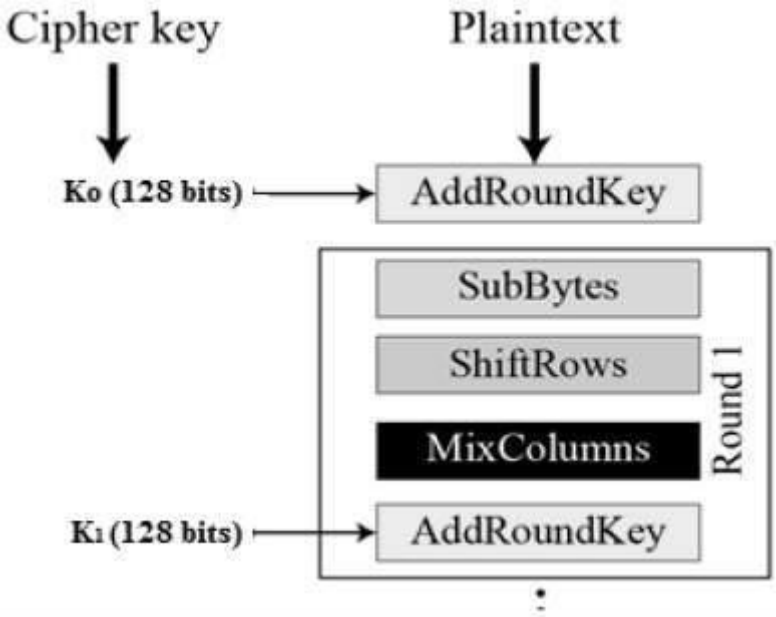


Figure II- 7: Processus du premier tour AES

II.5.1.2.3 Comparaison entre DES et AES

Depuis que l'algorithme DES a été rendu public, la communauté universitaire a mené des recherches et lancé des débats acharnés sur sa sécurité et d'autres aspects. Techniquement, la critique du DES s'est concentrée sur les domaines suivants [62] :

- En tant que chiffrement par blocs, l'unité de chiffrement de DES n'est que binaire 64 bits, ce qui est trop petit pour la transmission de données, car chaque bloc ne contient que 8 caractères, et certains de ces bits sont également utilisés pour la parité ou d'autres communications.
- Le nombre de clés DES est trop court, seulement 56 bits, et les clés utilisées à chaque itération sont générées récursivement, cette corrélation réduit inévitablement la sécurité du système cryptographique. La méthode pour trouver la clé est devenue possible.
- Le DES ne peut pas lutter contre la cryptanalyse différentielle et linéaire.

- La longueur de clé réelle utilisée par les utilisateurs DES est de 56 bits et la force de cryptage maximale théorique est de 256. Pour augmenter la force de chiffrement (telle que l'augmentation de la longueur de clé) de l'algorithme DES, la surcharge du système augmente de façon exponentielle. À l'exception de l'amélioration des fonctions matérielles et de l'augmentation des fonctions de traitement parallèle, la force de cryptage de l'algorithme DES ne peut pas être améliorée à partir de l'algorithme lui-même et de la technologie logicielle.

Par rapport à l'algorithme DES, l'algorithme AES résout sans aucun doute les problèmes ci-dessus.

L'algorithme AES se manifeste principalement sous les aspects suivants :

- Vitesse de calcul rapide.
- Les besoins en mémoire sont très faibles et adaptés aux environnements contraints.
- C’est un chiffrement itératif par blocs, avec des conceptions de bloc et de longueur de clé flexibles.
- La norme AES prend en charge une longueur de paquet variable. La longueur du paquet peut être définie sur n'importe quel multiple de 32 bits. La valeur minimale est de 128 bits et la valeur maximale est de 256 bits.
- La longueur de clé d'AES est supérieure à celle de DES.

Ci-dessous, nous avons résumé la différence entre DES et AES [62] :

	DES	AES
Date	1996	1999
Taille de bloc	64	128
La longueur de la clé	56	128, 192, 256
Nombre de tours	16	9, 11, 13
Encryptions primitives	Substitution, permutation	Substitution, shifts, bit, mixing
Primitives cryptographiques	Confusion, diffusion	Confusion, diffusion
Conception	Ouvert	Ouvert
Justification de la conception	Fermé	Ouvert
Processus de sélection	Secret	Secretmais accepter les commentaires publics ouverts
Vitesse de calcul	Plus rapide	Rapide
La sécurité	Faible	Élevé
Source	IBM amélioré par NSA	Cryptographes indépendants

Tableau II- 2: Comparaison entre DES et AES

II.5.2 Cryptographie à clé asymétrique

II.5.2.1 Introduction

La cryptographie asymétrique trouve son origine dans les travaux de recherche de Whitfield Diffie et Martin Hellma. Le papier qu'ils ont publié en 1976 a complètement révolutionné la cryptographie. L'algorithme qu'ils ont conçu évite aux communicants de s'entendre sur une clé partagée au préalable et a ainsi résolu le problème dit de distribution de clé rencontré avec la cryptographie symétrique [63]. Voyons un scénario pratique pour comprendre comment un tel système fonctionnerait. Supposons que Kawter souhaite envoyer un message à Chaima de manière confidentielle afin que personne d'autre que Chaima ne puisse comprendre le message, alors cela nécessiterait les étapes suivantes :

II.5.2.2 Principe de fonctionnement

Prenons l'exemple suivant [64]

Kawter l'expéditrice :

- Crypter le message en clair m en utilisant l'algorithme de cryptage E et la clé publique de Chaima pour préparer le texte chiffré C.
- $C = E(Puk_{Chaima}, m)$
- Envoyer le texte chiffré C à Chaima

Chaima la réceptrice :

- Déchiffrer le texte chiffré C en utilisant l'algorithme de déchiffrement D et sa clé privée Prk_{Chaima} pour obtenir le texte en clair d'origine m.
- $m = D(\mathrm{Prk}_{Chaima}, C)$

II.5.2.3 Illustration

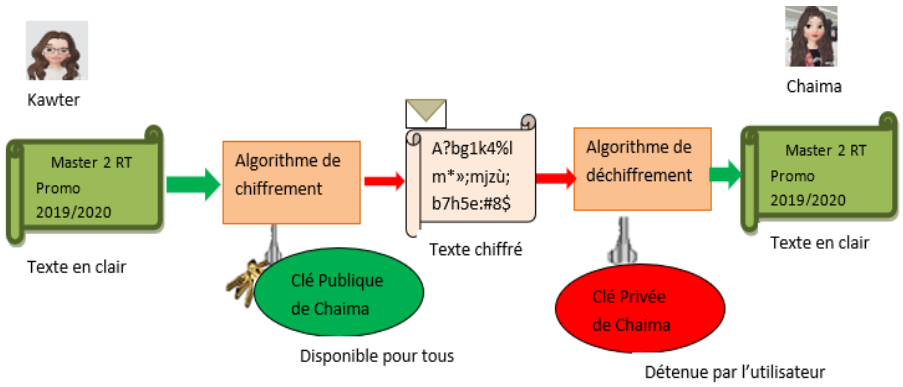


Figure II- 8: Cryptographie asymétrique pour la confidentialité

Notez que la clé publique doit être conservée dans un référentiel public accessible à tous et la clé privée doit être gardée comme un secret bien gardé. La cryptographie à clé publique fournit également un moyen d'authentification.

Le réceptrice, Chaima, peut vérifier l'authenticité de l'origine du message m de la même manière.

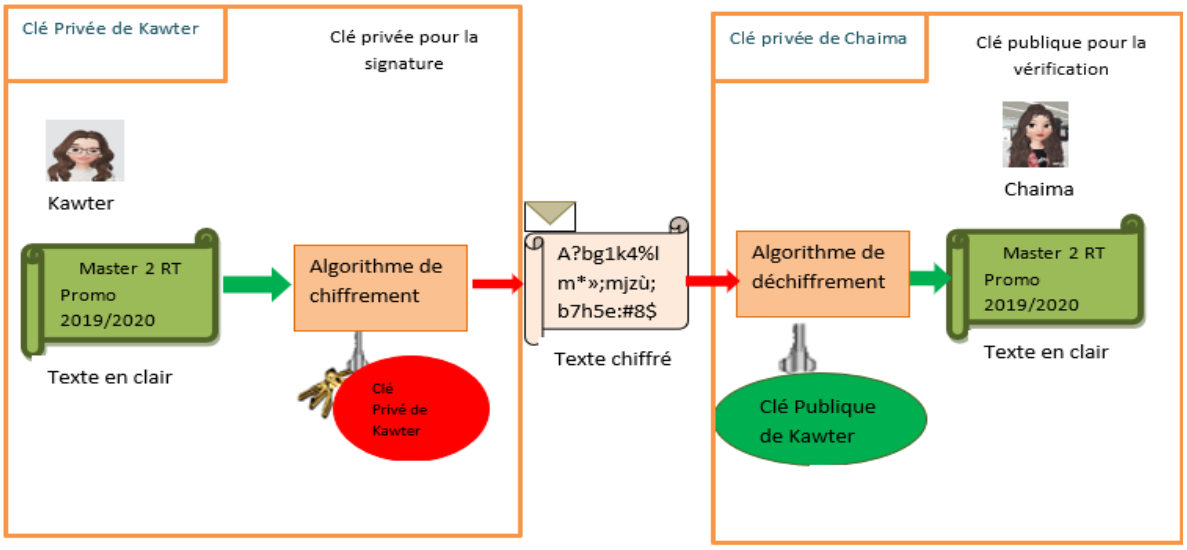


Figure II- 9: Cryptographie asymétrique pour l'authentification

Dans l'exemple de la figure II-17, le message a été préparé à l'aide de la clé privée de Kawter, afin de garantir qu'elle provienne uniquement de Kawter. Le message entier a servi de signature numérique. De sorte que la confidentialité et l'authentification sont souhaitables.

Le message doit d'abord être crypté avec la clé privée de l'expéditeur pour fournir une signature numérique. Ensuite, il doit être crypté avec la clé publique de réceptrice pour assurer la confidentialité. Il peut être représenté comme suit :

- $C = E[Puk_{Chaima}, E(Prk_{Kawter})]$
- $m = D[Puk_{Kawter}, D(Prk_{Chaima})]$

Comme vous pouvez le voir, le déchiffrement se produit dans son ordre inverse uniquement. Notez que la cryptographie à clé publique est utilisée quatre fois ici : deux fois pour le chiffrement et deux fois pour le déchiffrement. Il est également possible que l'expéditeur puisse signer le message en appliquant la clé privée à un petit bloc de données dérivé du message à envoyer, et non au message entier.

Nous avons examiné les utilisations des deux clés en cryptographie asymétrique, qui peuvent être résumées comme suit :

- Une clé publique, qui crypte les données, ou pour vérifier les signatures ;
- Et une clé privée ou secrète correspondante pour le décryptage ou pour créer des signatures ;

II.5.2.4 Les types d'algorithmes de chiffrement et déchiffrement

II.5.2.4.1 L'algorithme de chiffrement RSA

RSA est un algorithme de chiffrement mathématiquement significatif et important. Sa difficulté à craquer est basée sur la difficulté de décomposer un nombre composite en deux grands nombres premiers [65]. Parce que la factorisation du produit de deux grands nombres

premiers est assez difficile (la preuve est un peu compliquée et nécessite des concepts tels que la factorisation en nombres premiers, les nombres premiers et la congruence).

Le chiffrement dit asymétrique signifie que différentes clés peuvent être utilisées pour le chiffrement et le déchiffrement. De cette façon, la clé publique peut être envoyée à l'expéditeur et la clé privée est conservée sur lui-même et déchiffrée par la clé privée. Cela garantit que vous seul pouvez déchiffrer le message [66].

Le 12 décembre 2009, le RSA-768 (768 bits, 232 chiffres) a également été décomposé avec succès. À l'heure actuelle, la longueur de clé la plus sûre est RSA-1024 ou RSA-2048.Il s'agit actuellement de l'algorithme de cryptage le plus important de la planète [66].

II.5.2.4.1.1 Étapes pour la génération de clés

Nous utilisons un exemple pour comprendre l'algorithme RSA. Supposons que Kawter va avoir une communication cryptée avec Chaima. Comment peut-elle générer des clés publiques et privées ?

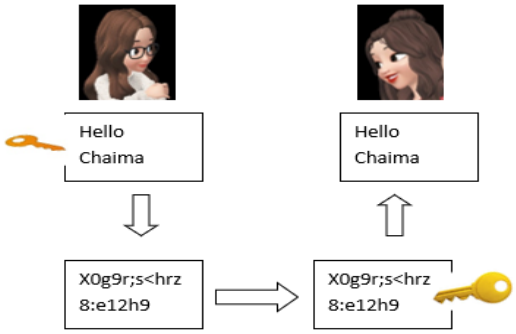


Figure II- 10: Étapes pour la génération de clés

Dans la première étape, deux nombres premiers inégaux p et q sont choisis au hasard. Chaima a choisi 61 et 53. En pratique, plus ces deux nombres premiers sont grands, plus il est difficile de se fissurer.

Dans la deuxième étape, le produit n de p et q est calculé.

Chaima a multiplié 61 et 53.

$$n = 61 \times 53 = 3233$$

Equation II- 1: Produit de n et p.

La troisième étape, consiste à calculer la fonction d'Euler ϕ (n) de n. Selon la formule

$$\Phi (n) = (p-1)(q-1)$$

Equation II- 2: Calcule de la fonction d'Euler

Chaima a calculé que ϕ (3233) est égal à 60 × 52, ce qui correspond à 3120.

La quatrième étape, consiste à choisir au hasard un entier e, à condition que 1 <e <ϕ (n), et e et ϕ (n) soient des nombres premiers.

Chaima avait choisi entre 1 et 3120 et a été choisie au hasard 17. En pratique, 65537 est souvent sélectionné.

Dans la cinquième étape, calculez l'élément inverse d de e pour ϕ (n).

Le soi-disant **"élément inverse modulaire"** signifie qu'il existe un entier d, de sorte que le reste de ed divisé par ϕ (n) est 1.

$$ed \equiv 1 \pmod{\phi(n)}$$

Equation II- 3: Calcule d'élément inverse d.

Par conséquent, trouver l'élément inverse d revient essentiellement à résoudre l'équation linéaire binaire suivante :

$$e.x + \phi(n).y = 1$$

Equation II- 4: Equation linéaire binaire

Étant donné e = 17, ϕ (n) = 3120,

$$17x + 3120y = 1$$

Cette équation peut être résolue en utilisant « **l'algorithme euclidien étendu** », et le processus spécifique est omis ici. En bref, Chaima a calculé un ensemble de solutions entières comme (x,y) = (2753, -15), c'est-à-dire d = 2753.

Jusqu'à présent, tous les calculs sont terminés.

Dans la sixième étape, n et e sont encapsulés dans une clé publique, et n et d sont encapsulés dans une clé privée.

Dans l'exemple de Chaima, n = 3233, e = 17 et d = 2753, la clé publique est donc (3233,17) et la clé privée est (3233, 2753).

II.5.2.4.1.2 Chiffrement et déchiffrement

Avec les clés publiques et secrètes, le chiffrement et le déchiffrement sont possibles.

(1) La clé publique (n, e) est utilisée pour le chiffrement :

Supposons que Kawter veuille envoyer des informations chiffrées m à Chaima, elle chiffrera m avec la clé publique de Chaima (n, e). Notez ici que m doit être un entier (la chaîne peut prendre des valeurs ascii ou Unicode), et m doit être inférieur à n.

Le soi-disant "cryptage" consiste à calculer c :$m^e = C \ mod(n)$

La clé publique de Chaima est (3233, 17), et le m de Kawter est supposé être 65, alors l'équation suivante peut être calculée : $65^{17} = 2790(mod3233)$

Donc, c'est égal à 2790, et Kawter envoie 2790 à Chaima.

(2) *La clé privée (n, d) est utilisée pour le déchiffrement :*

Après que Chaima a obtenu 2790 de Kawter, elle a utilisé sa clé privée (3233, 2753) pour la décrypter. Il peut être prouvé que l'équation suivante doit être vérifiée : $C^d = m \, mod(n)$

Autrement dit, le reste de la puissance de c divisé par n est m. Maintenant, c'est égal à 2790 et la clé privée est (3233, 2753), puis Chaima calcule : $2790^{2753} = 65 \, mod(3233)$

À ce stade, le processus complet de « **chiffrement-déchiffrement** » est terminé.

Nous pouvons voir que si nous ne connaissons pas d, il n'y a aucun moyen de trouver m à partir de c. Comme mentionné précédemment, savoir que d doit être décomposé en n, ce qui est extrêmement difficile à réaliser, donc l'algorithme RSA garantit la sécurité de la communication.

Vous pourriez demander que la clé publique (n, e) ne puisse chiffrer que des entiers m inférieurs à n, alors que faire si vous voulez chiffrer des entiers supérieurs à n ?

Il existe deux solutions : l'une consiste à diviser le message long en plusieurs messages courts, chacun étant chiffré séparément, l'autre à choisir d'abord un "algorithme de chiffrement symétrique" (tel que DES) et à utiliser la clé de cet algorithme, chiffrez les informations, puis chiffrez la clé DES avec la clé publique RSA.

La méthode de chiffrement à clé publique RSA offre également une authentification à l'aide d'une signature numérique. Notez ici qu'un algorithme différent appelé algorithme de signature numérique (DSA) peut également être utilisé dans de telles situations que nous verrons dans la section suivante. RSA est largement utilisé avec HTTPS sur les navigateurs Web, les e-mails, les VPNS et la télévision par satellite. En outre, de nombreuses applications commerciales ou les applications dans les magasins d'applications sont également signées numériquement à l'aide de RSA. SSH utilise également la cryptographie à clé publique ; lorsque vous vous connectez à un serveur SSH, il diffuse une clé publique qui peut être utilisée pour crypter les données à envoyer à ce serveur. Le serveur peut ensuite décrypter les données à l'aide de sa clé privée. L'algorithme de signature numérique DSA a été conçu par la NSA[26] dans le cadre de la norme de signature numérique DSS et normalisé par le NIST. Notez que son objectif principal est de signer les messages numériquement, et non le cryptage. Juste pour paraphraser, RSA est à la fois dédié pour la gestion des clés et l'authentification tandis que DSA est dédié uniquement pour l'authentification [67].

II.5.2.4.2 La signature numérique

Lorsqu'une personne envoie des données via un document, il devient important d'identifier son authenticité pour des raisons de sécurité et de sûreté. Des signatures numériques sont utilisées pour cette identification. L'authentification des documents signifie savoir qui les a créés et qu'ils n'ont pas interféré lors de leur transmission. Ces signatures sont créées à l'aide de certains algorithmes. L'algorithme de signature numérique (DSA) en

[26] **NSA :** National Security Agency est une agence de renseignement de niveau national du Département de la défense des États-Unis, placée sous l'autorité du directeur du renseignement national.

fait partie. DSA est un type d'algorithme **de chiffrement à clé publique** et il est utilisé pour générer une signature électronique [68].

II.5.2.4.2.1 Introduction

Les signatures numériques sont idéales pour l'intégrité et la non-répudiation. Une signature numérique est une combinaison de hachage et de chiffrement asymétrique. Autrement dit, un message est d'abord haché, et ce hachage est chiffré avec la clé privée de l'expéditeur. Cela constitue la signature, qui est envoyée avec le message.

II.5.2.4.2.2 Fonctionnement d'algorithme de la signature numérique

Il s'agit d'un algorithme de chiffrement à clé publique conçu pour créer une signature électronique. Une signature est créée « en privé » mais peut être vérifiée « en public ». En d'autres termes, il n'y a qu'un seul sujet qui peut créer une signature ajoutée à un message, mais n'importe qui est en mesure de vérifier si la signature est correcte ou non.

II.5.2.4.2.3 Schéma fonctionnel de la signature numérique

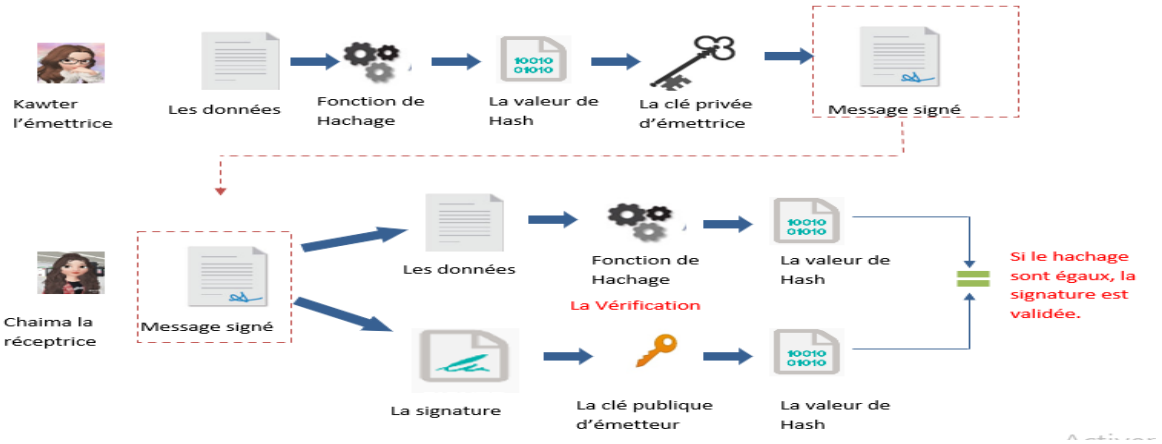


Figure II- 11: Schéma représente comment la signature fonctionne

II.5.2.4.2.4 Explication du schéma fonctionnel

Premièrement, chaque personne adoptant ce schéma possède une paire de clés publique-privée en cryptographie [69].

Les paires de clés utilisées pour le chiffrement ou le déchiffrement et la signature ou la vérification sont différentes pour chaque signature. Ici, la clé privée utilisée pour la signature est appelée clé de signature et la clé publique comme clé de vérification dans cet algorithme.

Ensuite, les gens prennent le signataire qui alimente les données à la fonction de hachage et génère un hachage des données de ce message.

Maintenant, la valeur de hachage et la clé de signature sont ensuite transmises à l'algorithme de signature qui produit la signature numérique sur un hachage donné de ce message. Cette signature est ajoutée aux données, puis les deux sont envoyées au vérificateur pour sécuriser ce message.

Ensuite, le vérificateur alimente la signature numérique et la clé de vérification dans l'algorithme de vérification de ce DSA. Ainsi, l'algorithme de vérification donne une certaine valeur en sortie sous forme de texte chiffré.

Ainsi, le vérificateur exécute également la même fonction de hachage sur les données reçues pour générer une valeur de hachage dans cet algorithme.

Maintenant, pour la vérification, la signature, la valeur de hachage et la sortie de l'algorithme de vérification sont comparées avec chaque variable. Sur la base du résultat de la comparaison, le vérificateur décide si la signature numérique est valide pour cela ou non.

Par conséquent, la signature numérique est générée par la clé privée du signataire et personne d'autre ne peut avoir cette clé pour sécuriser les données. Le signataire ne peut pas répudier la signature des données à l'avenir pour sécuriser ces données par la cryptographie.

II.5.2.4.2.5 Importance de la signature numérique

Par conséquent, toute analyse cryptographique de la signature numérique utilisant la cryptographie à clé publique est considérée comme un outil très important ou principal et utile pour assurer la sécurité des informations en cryptographie en cryptanalyse [67].

Ainsi, outre la possibilité de fournir la non-répudiation du message, la signature numérique fournit également l'authentification du message et l'intégrité des données en cryptographie.

Ceci est réalisé par la signature numérique qui réalise :

- ***Authentification des messages :*** Par conséquent, lorsque le vérificateur valide la signature numérique à l'aide de la clé publique d'un expéditeur, il est assuré que la signature n'a été créée que par un expéditeur qui possède la clé privée secrète correspondante et que personne d'autre ne le fait par cet algorithme.

- **Intégrité des données :** en fait, dans ce cas, un attaquant a accès aux données et les modifie, la vérification de la signature numérique à l'extrémité du récepteur échoue dans cet algorithme. Ainsi, le hachage des données modifiées et la sortie fournie par l'algorithme de vérification ne correspondent pas à la signature de cet algorithme. Maintenant, le récepteur peut refuser le message en toute sécurité en supposant que l'intégrité des données a été violée pour cet algorithme.

- **Non-répudiation** : Par conséquent, c'est juste un nombre que seul le signataire connaît la clé de signature, il ne peut que créer une signature unique sur une donnée de ce message pour changer de cryptographie. Ainsi, le destinataire peut présenter les données et la signature numérique à un tiers comme preuve si un différend survient à l'avenir pour sécuriser les données.

II.5.2.4.3 EllipticCurve Digital Signature Algorithm

ECDSA c'est in algorithme de chiffrement à courbe elliptique pour apprendre la signature numérique d'un algorithme [70].

II.6 Comparaison entre cryptographie symétrique et asymétrique

Le chiffrement asymétrique utilise deux types de clés, tandis que le chiffrement symétrique utilise un seul type de clé. Par conséquent, le traitement est plus simple dans ce dernier cas, et les deux parties échangeant des données utilisent la même clé secrète pour coder et décoder les données reçues [71].

Le chiffrement symétrique est depuis longtemps la norme en cryptographie, mais il existe plusieurs failles dans la sécurité des communications qui ont créé une cryptographie asymétrique. Cependant, le cryptage symétrique est encore principalement utilisé pour transmettre de grandes quantités de données.

La cryptographie à clé symétrique est largement utilisée pour crypter des données telles que des messages et des fichiers car le cryptage et le décryptage peuvent être effectués à grande vitesse. La cryptographie à clé publique présente l'inconvénient que le cryptage / décryptage est plus lent que la cryptographie à clé symétrique. Lors de l'échange de messages par e-mail, etc., la vitesse élevée de la méthode de cryptage à clé commune et la commodité de la méthode de cryptage à clé publique sont combinées, le message est crypté avec le cryptage à clé commune et la clé de cryptage est cryptée avec le cryptage à clé publique.

II.7 Fonction de hachage

Le « hachage » est un concept qui se confond facilement avec le chiffrement. Le hachage est une technologie qui convertit les données d'origine en données de longueur fixe en effectuant un traitement à l'aide d'une fonction de hachage. Il peut être converti, mais ne peut pas être annulé. En d'autres termes, il n'y a pas d'équivalent au "décryptage" dans le hachage [72].

Le hachage a pour but de gérer les mots de passe et de vérifier les fichiers. Le hachage et le stockage des mots de passe éliminent la nécessité de gérer les mots de passe bruts. Pour vous authentifier, il vous suffit de comparer la version hachée du mot de passe entré avec le mot de passe haché stocké. En outre, lorsque vous téléchargez un fichier volumineux sur Internet, le hachage est utilisé pour comparer le fichier sur le serveur avec le fichier téléchargé. Il est difficile de comparer des fichiers volumineux tels qu'ils sont, nous les hachons donc et les comparons [73].

Les fonctions de hachage incluent "**MD5**", "**SHA-1**" et "**SHA-256**". Cependant, MD5 a longtemps été souligné comme étant vulnérable. Nous avons mentionné précédemment qu'il n'y a pas d'équivalent au « déchiffrement » dans le hachage, mais MD5 à une vulnérabilité où les données d'origine avec une valeur de hachage spécifique sont calculées. Une attaque qui exploite une telle vulnérabilité est appelée attaque par collision. Par conséquent, l'utilisation de MD5 n'est pas recommandée à l'heure actuelle. L'utilisation d'une fonction de hachage permet de s'assurer de l'intégrité des données et indirectement de les authentifier. Il existe bien sûr de nombreuses autres applications pour les fonctions de hachage, comme les MACs, certificats, etc [73].

II.7.1 MD5

MD5 est une fonction de hachage largement utilisée dans le domaine de la sécurité informatique pour assurer la protection de l'intégrité des messages.

MD5 est largement utilisé pour l'authentification par mot de passe et l'identification des clés de divers logiciels. MD5 utilise une fonction de hachage, et son application typique est de générer un résumé de message pour une information pour éviter toute falsification. Une application typique deMD5est de générer des empreintes digitales sur un message pour éviter qu'il ne soit "falsifié". S'il existe un organisme de certification tiers, MD5 peut également empêcher l '"autorité" de l'auteur du document, qui est la soi-disant application de signature numérique. MD5 est également largement utilisé pour l'authentification de connexion des systèmes d'exploitation, tels qu'UNIX [74].

II.7.2 SHA-1

SHA1 est un algorithme de résumé de message aussi populaire que MD5. Pour les messages de moins de 64bits, SHA1 génère un résumé de message de160bits. Lorsqu'un message est reçu, ce résumé de message peut être utilisé pour vérifier l'intégrité des données. SHA1 ne peut pas récupérer les informations du résumé de message, et deux messages différents ne généreront pas le même résumé de message. De cette façon, SHA1 peut vérifier l'intégrité des données, alors on dit que SHA1est une technologie pour assurer l'intégrité des fichiers [75].

SHA1 est un algorithme plus sécurisé que MD5. En théorie, tous les algorithmes de vérification numérique qui utilisent la méthode "message digest" ont des "collisions", c'est-à-dire que les résumés de messages calculés par deux choses différentes sont les mêmes et interopérables. La carte de triche est juste cela. Cependant, il est très difficile pour un algorithme de haute sécurité de trouver la "collision" des données spécifiées, et il est encore plus difficile de calculer la "collision" à l'aide d'une formule. Jusqu'à présent, seul MD5 a été craqué dans les algorithmes de sécurité généraux.

II.7.3 Comparaison entre MD5 et SHA

La comparaison est conclue dans le tableau suivant [76] :

SHA	MD5
SHA représente l'algorithme de hachage sécurisé	MD5 signifie Message Digest
SHA est une famille de fonctions de hachage cryptographiques développées par le NIST[27]	Il s'agit d'une fonction de hachage cryptographique largement utilisée qui produit une valeur de hachage de 128 bits
Des versions plus sécurisées de SHA-1 sont disponibles telles que SHA-256, SHA-384 et SHA-512	On pense que MD5 est cryptographiquement cassé et peut avoir des collisions
La version optimisée de SHA-1 est plus rapide que MD5	MD5 est relativement plus rapide que SHA
Il n'y a eu aucune attaque sérieuse contre SHA-1	D'autres attaques ont été signalées sur MD5

Tableau II- 3: Comparaison entre MD5 et SHA

II.8 La longueur de la clé

Le nombre de bits utilisés pour le chiffrement ou le déchiffrement est appelé longueur de clé. Des clés plus longues augmentent la sécurité, mais présentent l'inconvénient d'un chiffrement et d'un déchiffrement plus lents [77].

De plus, la longueur de la clé varie en fonction de la méthode de cryptage. La longueur de clé utilisée dans la cryptographie à **clé symétrique** est d'environ 40 à 128 bits. Dans la cryptographie à **clé publique**, des clés de différentes longueurs d'environ 100 bits à 2048 bits sont utilisés.

II.9 Limite de la cryptographie [78] [79]

II.9.1 Limitation de la cryptographie a clé publique (asymétrique)

Dans le chiffrement asymétrique, l'utilisateur utilise la même clé pour chiffrer et déchiffrer, La raison principale que la cryptographie à clé publique a une limitation réside sur la performance et la lenteur à laquelle se font les opérations de chiffrement et de déchiffrement. En effet cette méthode de cryptage nécessite un nombre très important de calculs c'est-à-dire le temps de calcul devient long parce que la lenteur d'exécution nécessité de faire beaucoup de calculs.

[27] **NIST** : National Institute of Standards and Technology est un laboratoire de sciences physiques et une agence non réglementaire du Département du commerce des États-Unis. Sa mission est de promouvoir l'innovation et la compétitivité industrielle.

De plus, la cryptographie à clé publique présente un autre problème, à savoir que la taille des données cryptées est limitée. Par exemple, RSA ne peut pas crypter des données supérieures à 4096 bits. Cette taille est liée à la longueur de sa clé c'est-à-dire Cette méthode du chiffrement ne peut donc pas être utilisée pour des grands fichiers à transférer.

II.1 Limitation de la cryptographie a clé secrètes (symétrique)

Dans le chiffrement symétrique, c'est une même clé secrète qui permet à la fois d'effectuer le chiffrement et le déchiffrement. Lorsque la personne qui devra déchiffrer les données n'est pas la même que celle qui les a chiffrées, plusieurs problèmes se posent : Tout d'abord, il faut une nouvelle clé secrète pour chaque couple émetteur/récepteur. De plus, expéditeur et destinataire doivent se mettre d'accord sur une clé secrète et de l'échanger de façon confidentielle, c'est-à-dire le grand problème principal c'est la contribution des clés. En effet, la clé de chiffrement est identique à la clé de déchiffrement. Ainsi, c'est la même clé qui va nous permettre à la fois de crypter le message et aux destinataires de le décrypter. Donc, le problème majeur est l'échange de clé entre les deux individus. Or, cela est difficile à réaliser, puisque, tant que la clé n'est pas transmise, il n'existe pas de moyen sûr d'échange d'information, à part une rencontre physique.

Partie 3 : Fonctionnement de la blockchain

II.10 Introduction

Le but de cette étude est d'analyser la sécurisation de la blockchain. Pour la première fois, la technologie blockchain a été utilisée dans le secteur financier, où elle a servi de base à la création de la devise critique Bitcoin. Récemment, de plus en plus d'applications sont apparues qui étendent la fonction clé de cette technologie - le stockage décentralisé des données de transaction. Les transactions de ce type supposent que chaque membre du réseau peut effectuer une transaction directement avec tout autre membre du réseau sans impliquer un intermédiaire tiers. Les transactions ne sont plus stockées dans une base de données centralisée, mais transmises aux ordinateurs (appelés aussi les ressources informatiques) de tous les participants au réseau qui stockent les données localement suite à une phase d'enrôlement. Ces ressources sont couramment appelées **des nœuds** du fait qu'elles sont mises en réseau au travers d'internet.

Il existe deux types de nœuds [80] :

- Les **nœuds réguliers** : la plupart de ces nœuds fournit une capacité informatique ordinaires, à partir de laquelle les personnes peuvent émettre des demandes de transactions ;
- Les **nœuds « miners »** ou **mineurs** : Ce nœud est doté de grosse capacité de traitement, ces transactions sont regroupées dans un bloc qui va être validé par les mineurs du réseau en résolvant un problème mathématique complexe.

II.11 La phase d'enrôlement dans la blockchain

Pour participer aux blockchain une personne doit enrôler un de ses équipements informatiques comme un nœud de la blockchain. Au cours de cette opération, il faut investir

dans du matériel et des programmes de minage spécialisés. Ces programmes de minage (logiciels) ne sont pas directement liés à Blockchain Core et sont exécutés en parallèle pour essayer de miner des blocs. Ce logiciel est personnalisé avec un numéro de compte blockchain. Il est impératif que le nœud à une propriété de conserver le logiciel téléchargée et le mot de passe qui lui permettre de déverrouiller la clé privée, sinon il perd l'accès à son compte blockchain et ne pourra plus faire des transactions sur ce compte [80][81].

II.12 La phase de transaction

Dans le chapitre précédent, nous avions une introduction rapide à la technologie blockchain. Dans cette partie du chapitre, nous allons voir ce qui se passe derrière une transaction bitcoin.

> Remarque : C'est une idée fausse courante que la blockchain et les bitcoins sont une seule et même chose. Cependant, les bitcoins ne sont qu'une implémentation de la technologie blockchain. Il en va de même pour les autres monnaies numériques.

Prenons un Exemple simple : Kawter et ses amies sont allées au restaurant où les paiements en bitcoins sont acceptés. Après toute la soirée à manger, rigoler et à s'amuser, il est temps de payer la facture !

Le serveur sort un smartphone et montre le code QR à Kawter pour le paiement de la facture. Kawter sort également son smartphone, ouvre son application bitcoin, scanne le code QR, vérifie si les détails du paiement sont corrects et appuie sur le bouton PAYER.

Le **QR code** est une sorte de code barre mais plus complexe, il contient plusieurs types d'informations, comme :

- Le **portefeuille électronique du récepteur** (dans notre cas restaurant)
- **Montant** du bitcoin à transférer
- **Informations générales** (comme le nom du destinataire)

Figure II- 12: Code QR

Lorsque Kawter appuie sur le bouton PAYER ;

- Elle crée une nouvelle transaction qui contient les détails du portefeuille électronique de l'expéditrice (Kawter) et du portefeuille électronique du destinataire (le restaurant).

Ce qui se passe derrière la scène ?

En bref, la blockchain est comme un grand livre numérique qui est distribué sur de nombreux ordinateurs et ajoute continuellement des enregistrements comme dans une file

d'attente. Son réseau est similaire à BitTorrent (P2P), dans lequel le transfert se fait entre les utilisateurs plutôt qu'un serveur central puissant [82].

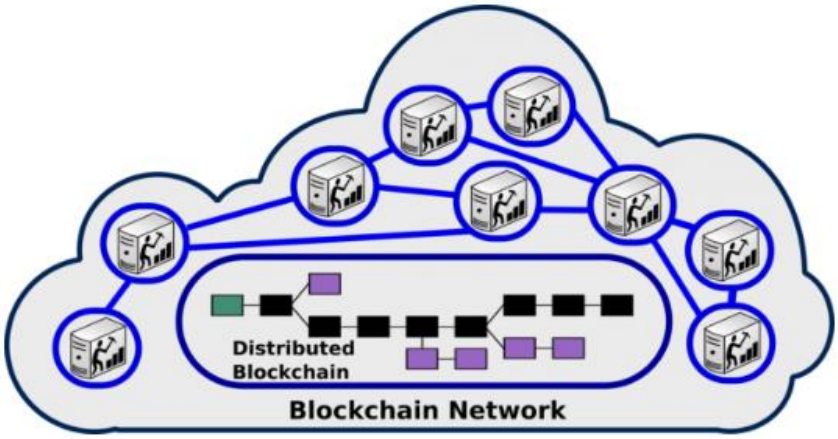


Figure II- 13: Blockchain distributée

La figure suivante illustre le fonctionnement de la blockchain lorsqu'on a effectué une transaction :

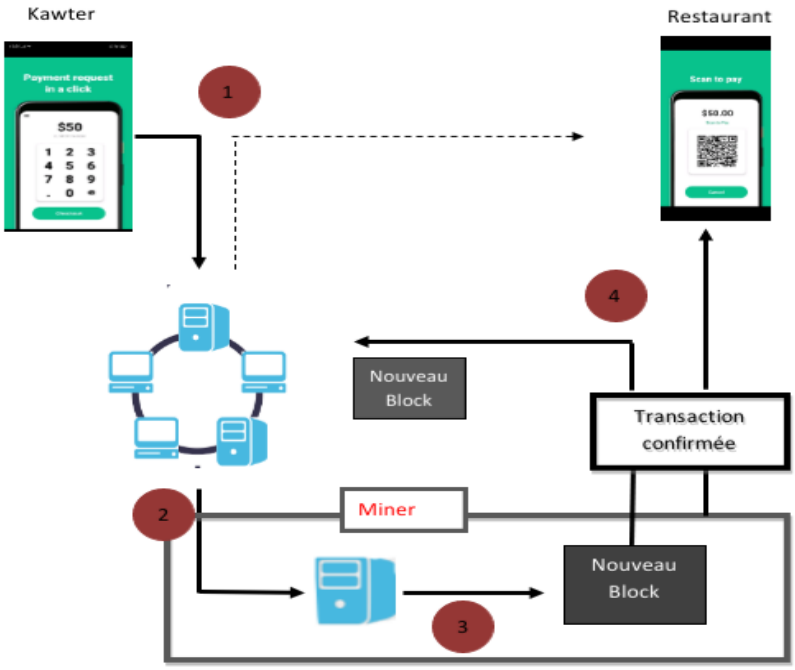


Figure II- 14: Fonctionnements de la blockchain

La transaction se propage dans le réseau bitcoin comme suite :

La **phase 1** du schéma II-14 ; le serveur de restaurant peut voir le paiement de Kawter sur son téléphone presque instantanément, le statut n'est cependant **PAS CONFIRMÉ**.

- *Qui décide si une transaction est valide ou non ?*

La transaction est considérée comme valide lorsque des nœuds spéciaux du réseau (appelés Miner) ajouteront cette transaction dans un bloc de la blockchain.

Ce processus est appelé Mining (**phase 2** du graphique) et il a les objectifs suivants :

- Pour **valider la transaction** (en se référant au protocole de consensus) ou pour rejeter une transaction invalide.
- Pour créer de nouveaux bitcoins.

Les mineurs qui agglomérant sous forma d'un bloc les transactions valides est tentent de valider le bloc par la résolution d'un problème mathématique complexe appelé Proof of Work (PoW). Ce travail de résolution de problème s'appelle « minage ». Le mineur qui a terminé le minage en premier diffuse sa solution à tous les nœuds qui vérifient la preuve PoW associée. En cas de validité, chaque nœud ajoute le bloc dans la blockchain et les mineurs commencent à miner le bloc suivant (**phase 3** de la figure II-14).

Le fait d'inscrire massivement un bloc dans la blockchain signifie qu'un consensus a été atteint parmi les nœuds [82].

Enfin, le serveur de restaurant voit dans son portefeuille électronique que le paiement a été confirmé. Tout ce processus dure environ 10 minutes.

II.13 Qu'est-ce qu'une transaction ?

Maintenant, au lieu d'utiliser une simple chaîne de caractères dans la data, nous allons-y insérer une transaction ! Nous avons des portefeuilles, il est temps de créer le processus pour envoyer de l'argent d'un portefeuille à un autre. Ce processus est appelé une transaction. Dans un premier temps, pour effectuer une transaction, nous aurions besoin de

- Adresse du portefeuille de l'expéditeur.
- Adresse du portefeuille du récepteur.
- Montant à envoyer.

Prenons un aperçu d'un ajout d'une nouvelle transaction dans la chaîne de blocs Bitcoin [83][84].

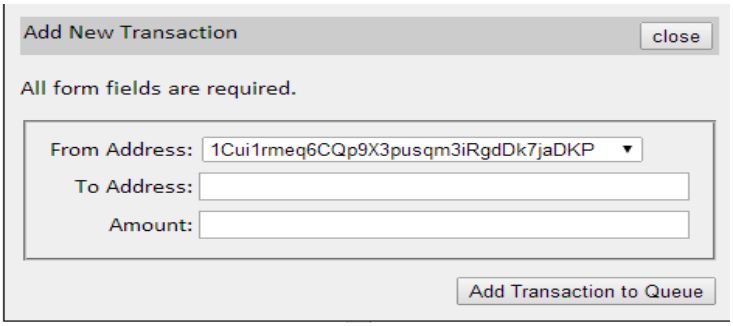


Figure II- 15: New transaction

La transaction aura notre clé publique en tant qu'émetteur, et la clé publique du destinataire, et nous signons cela avec notre clé privée. Voir figure ci-dessous :

Figure II- 16: Signature de notre transaction avec notre clé privée

La blockchain est maintenant capable d'utiliser notre clé publique pour vérifier que nous sommes bien à l'origine de cette transaction (fig. ci-dessous), que nous sommes bien l'expéditeur et que la transaction n'a pas été altérée par un tiers.

Figure II- 17: Blockchain vérifie que cette transaction a bien été envoyée par le propriétaire de la clé privée

Donc une transaction doit nécessairement contenir :

- *TxID :* un identifiant unique de transaction avec une fonction de crypto sha256 utilisées pour identifier d'une façon exclusive une transaction.
- *Input :* l'adresse de puis laquelle la transaction est émise.
- *Output :* les adresses réceptrices des quantités échangées.

Dans la transaction ci-dessus, nous pouvons remarquer les propriétés suivantes [83][84]:

Hacher	Cela fonctionne comme la clé unique de cette transaction. C'est une combinaison de données incluses dans la transaction qui crée cette clé.
Apparait dans	Dans quel bloc se trouve cette transaction.
Nombres d'entrées	Le nombre d'entrées. Les entrées sont des références à des transactions dont le portefeuille, qui souhaite envoyer de l'argent, a reçu de l'argent.
Total BTC en	Le nombre total de Bitcoins reçus par le portefeuille.
Nombre de sorties	Le nombre d'expéditeurs.
Total BTC out	Le nombre total de Bitcoins envoyés par le portefeuille. Il doit être égal au « Total BTC in ». Si vous souhaitez envoyer seulement une partie du montant de votre portefeuille, vous pouvez envoyer la partie à la destination et le reste à vous-même.
Taille	Taille des données en octets
Frais	Si vous souhaitez donner une petite récompense au bon ami qui a pris votre transaction, l'ajoutez dans un bloc et trouvez un nonce pour ce bloc afin qu'il puisse être ajouté dans la chaîne de blocs.

Tableau II- 4: Propriété d'une transaction

L'illustration suivante montre un exemple de transaction qui prend 4 entrées et génère deux sorties (à l'exclusion de la sortie de commission) :

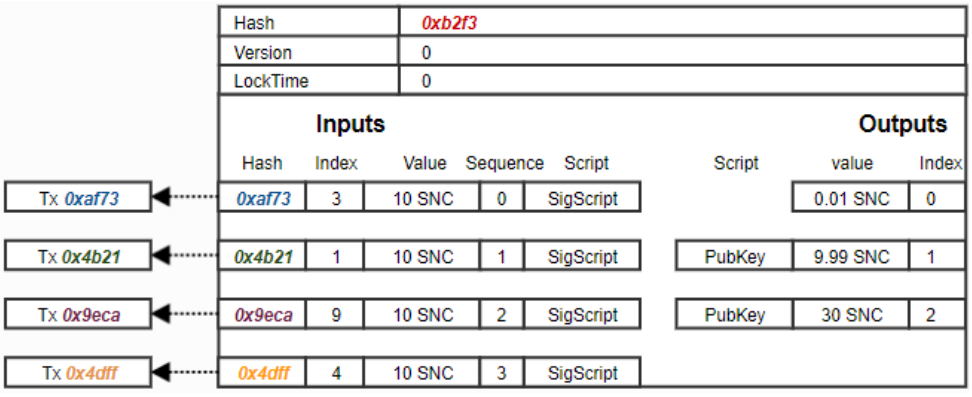


Figure II- 18: Exemple de transaction

- *Entrée de transaction :* Les entrées de transaction font référence à une sortie de transaction utilisable (UTXO) capturée dans la structure de données de point d'entrée de transaction. De plus, un script doit être ajouté avec une signature valide permettant à la partie qui crée la transaction de réclamer l'UTXO référencé [83][84].

Taille	Nom	Type	Commentaire
44	PreviousOutput	TransactionInputOutpoint	Référence à UTXO
4	Séquence	uint 32	L'index de l'entrée spécifique dans la transaction
var.	Scénario	[] octet	Signature pour vérifier la propriété de la clé publique de l'UTXO référencé

Tableau II- 5: Structure de données Transaction Input

- *Point d'entrée de transaction* **:** Les points d'entrée de transaction sont des références aux sorties de transaction précédentes. La référence est capturée avec un hachage de transaction, un index de l'UTXO à réclamer et la valeur contenue dans l'UTXO [83][84].

Taille	Nom	Type	Commentaire
32	Hacher	[32] octet	Le hachage de la transaction référencée
4	Indice	Uint 32	L'index de la sortie spécifique dans la transaction
8	Valeur	uint64	Valeur UTXO

Tableau II- 6: Structure de données Transaction Input Outpoint

- *Sortie de transaction* [83][84]

Taille	Nom	Type	Commentaire
4	Indice	Uint 32	Paramètre de commande
8	Valeur	Uint 64	Valeur de sortie de transaction
var.	Scénario	[] octet	Script définissant les conditions pour revendiquer cette sortie
32	NodeID	[32] octet	Clé publique du nœud qui souhaite participer au PoS

Tableau II- 7: Structure de données Transaction Output

II.14 Qu'est-ce qu'un bloc ?

Comme indiqué précédemment, une blockchain est une chaîne de blocs contenant chacun plusieurs transactions, et qui vont être inscrits au fur et à mesure dans la blockchain par des nœuds du réseau.

Figure II- 19: Chaine de bloc

Tout d'abord, la première compréhension de la blockchain est qu'il s'agit qu'un bloc regroupe un ensemble de transaction et vise à cristalliser le contenue des transactions et du bloc, et la position du bloc dans la blockchain. Cette cristallisation repose sur deux procédés essentiels. La première, les fonctions de hachage et la deuxième l'arbre de Merkle permettent de rigidifier la structure de transaction et de blocs en venant souder tous ces éléments entre eux [80].

Le premier bloc d'une blockchain est appelé "**Genesis Block**" et est le seul qui ne contient pas les données des blocs précédents, car il n'y en a aucun avant lui. Comme le montre la figure suivante :

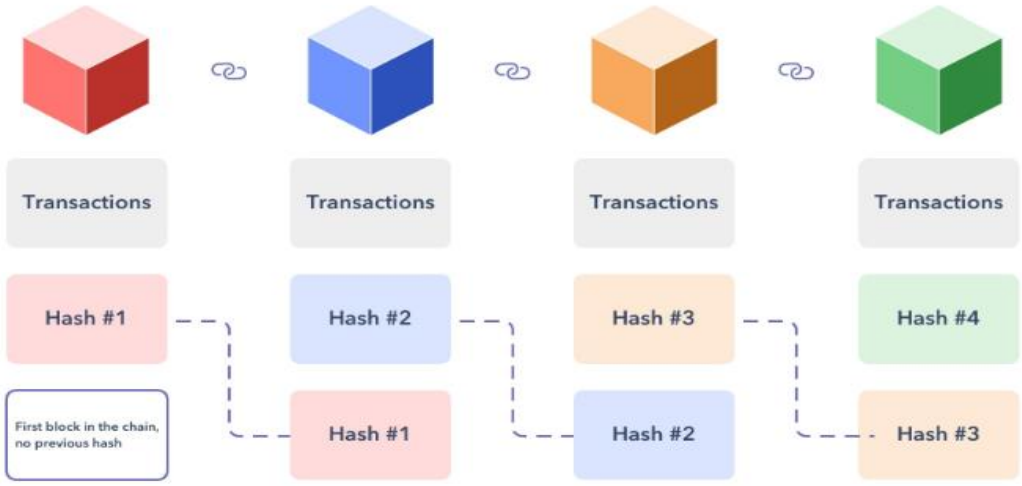


Figure II- 20: Genesis Block

L'implémentation peut différer d'une blockchain à l'autre, mais les principaux éléments d'un bloc sont les suivants

Un bloc contient trois éléments comme le représente l'illustration ci-dessous :

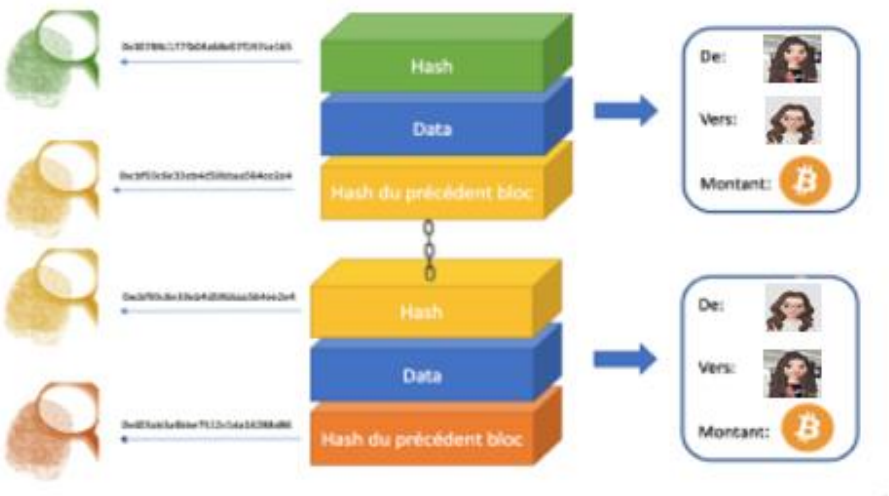


Figure II- 21: Qu'est-ce qu'un bloc et que contient-il En général ?

- ***Un « hash » :*** l'empreinte digitale du bloc. C'est une suite de caractères unique (comprenant chiffres et lettres) qui permet d'identifier avec précision un bloc.
- ***Des données :*** par exemple les données d'une transaction en Bitcoin, incluant l'adresse du porte-monnaie virtuel du payeur, celle du receveur, ainsi que le montant de l'opération.
- ***Le hash du bloc précédent :*** l'empreinte digitale du bloc précédent. C'est ce qui permet de situer le bloc dans la chaine.

II.14.1 La structure d'un bloc

Un bloc est composé d'un en-tête et d'un certain nombre de transactions [85].

L'en-tête du bloc contient un hash – résultat d'une fonction de hachage – de l'en-tête du bloc qui le précède c'est la référence que nous évoquions précédemment et qui assure un lien immuable entre les blocs.

a) *L'entête de bloc (block header) :* L'entête contient ce qu'on appelle les métadonnées à propos du block concerné. On peut répertorier principalement 3 ensembles de métadonnées :
   - Le hash du block précédent : dans une blockchain, chaque bloc est l'héritier du bloc précédent. Cela est dû au fait qu'il utilise le hash du block d'avant pour créer son propre hash.
   - Les informations relatives au minage : on peut y retrouver la date exacte où le bloc a été créé, la difficulté de minage actuel, les personnes qui ont été récompensées pour avoir validé le bloc, etc.
   - La racine de l'arbre de Merkle : il s'agit d'un arbre structurant les données présentes dans le bloc.

b) *L'identificateur de bloc (block indentifiers) :* Afin d'identifier un bloc, celui-ci possède une signature digitale, appelée hash cryptographique. Il est possible d'identifier un block en connaissant sa position dans la blockchain. Par exemple, le block 452.525 signifie qu'il y eu avant lui 452.524 blocs.
L'arbre de Merkle (Merkletree) *:* Toutes les transactions sont agencées dans une structure que l'on appelle arbre de Merkle. Prenons l'exemple d'un block contenant 16 transactions et utilisons des lettres pour les identifier. Soient de A à P. L'algorithme de hachage donnera tout d'abord un hash pour chacune de ces transactions.

c) Ensuite, on combinera le hash de A et B pour avoir le hash de AB. Après, on combinera le hash de AB et CD pour obtenir le hash d'ABCD et ainsi de suite jusqu'à obtenir un hash pour ABCDEFGHIJKLMNOP. Ce dernier hash est appelé racine de Merkle.

Un bloc contient les transactions qui ont été traitées par les mineurs et fait partie intégrante de la blockchain une fois que celui-ci a été validé par l'ensemble du réseau. A partir de ce moment-là, les informations incluses dans ce dernier deviennent inaltérables. Chaque bloc possède un entête, un identificateur et un arbre de Merkle qui agence les transactions à l'intérieur de celui-ci.

La figure ci-dessous illustre les différents éléments d'un bloc.

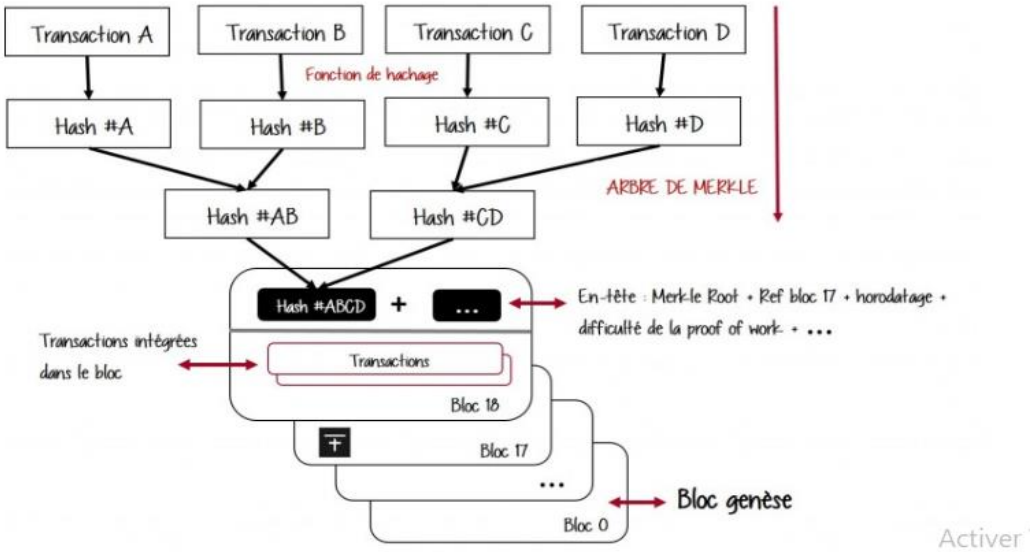


Figure II- 22: Contenus des blocs de la blockchain et arbre de Merkle

II.14.2 Les mineurs

L'exploration de crypto-monnaie est un processus de calcul pour enregistrer les données envoyées ou reçues par la crypto-monnaie sur la blockchain. Les gens qui font cette exploitation sont appelés "mineurs".

II.14.2.1 Qu'est-ce qu'un mineur ?

Les mineurs sont des acteurs importants dans les crypto-monnaies. Vous serez positionné en tant que technicien pour former des blocs tels que la preuve de travail (POW), et si vous réussissez dans l'exploitation minière, vous serez récompensé [86].

Les mineurs ont une grande puissance de hachage et effectuent des travaux de calcul. Sans eux, la technologie de la chaîne de blocs ne serait pas possible. Le minage effectué par le mineur garantit la crédibilité de la crypto-monnaie.

Les mineurs s'investissent dans l'équipement minier et approuvent les transactions, et si l'exploitation réussit, ils pourront gagner des récompenses minières.

Le minage calcule les données de transaction et, une fois approuvé, il crée une nouvelle blockchain. La blockchain garde une trace de toutes les transactions depuis la création de la monnaie virtuelle.

Les mineurs peuvent obtenir gratuitement la monnaie virtuelle calculée, au lieu de calculer les données de transaction effectuées par des tiers. Tous les mineurs ne peuvent pas obtenir de crypto-monnaies, et les récompenses sont soumises à certaines conditions.

Ceci est similaire à l'extraction du minerai d'une mine. C'est pourquoi on l'appelait "minage" ou "mineur".

II.14.2.2 Que fait l'exploitation minière par les mineurs ?

L'exploitation minière est basée sur la vision du monde selon laquelle le commerce est effectué sans organisation centrale dans le monde de la monnaie virtuelle. Personne ne garantit la crédibilité des transactions qui auraient lieu en l'absence d'une autorité centrale. Nous effectuons de l'exploitation minière pour garantir sa crédibilité. L'exploitation minière fait référence à la tâche d'approuver et de confirmer les transactions.

Nous vérifions en fait les transactions en monnaie virtuelle et écrivons les informations sur les remises dans un registre des transactions (blockchain). Les informations de remise vérifient si la personne qui a réellement envoyé la crypto-monnaie est le titulaire de la crypto-monnaie et s'il existe des transactions en double. Pour ce faire, vous devez résoudre un grand nombre de problèmes de calcul. La difficulté de ce problème de calcul est si élevée que vous avez besoin d'un ordinateur de haute qualité. Bien sûr, il n'y a pas qu'un seul mineur, donc plusieurs personnes résolvent le problème de calcul à la fois. Par conséquent, la première personne qui souhaite réussir dans le secteur minier sera rémunérée.

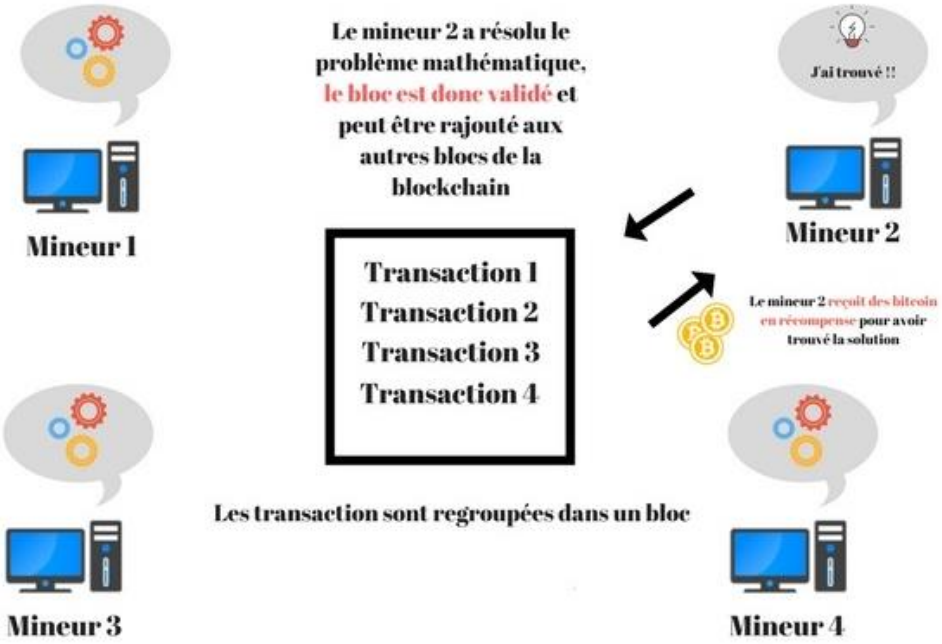


Figure II- 23: Validation d'un bloc de transactions par les mineurs

II.14.2.3 Comment fonctionne l'exploitation minière ?

Nous avons mentionné plus tôt que l'exploitation minière est une tâche de calcul, mais ce n'est pas seulement une tâche de calcul. En raison de la structure de la blockchain, il est nécessaire de connecter les blocs, mais ce qui relie ces blocs est appelé « **valeur de hachage** » [87].

Cette valeur de hachage est une valeur numérique créée à l'aide d'une formule de 0 à f. Cette valeur de hachage à la propriété que la valeur de sortie ne peut être prédite par aucun moyen et ne peut donc pas être altérée.

En multipliant cette valeur de hachage par une fonction de hachage, une chaîne de caractères appelée nonce sort par calcul. Si ce nonce est correct, l'extraction réussira. Le secteur minier recherche simplement ce nonce. Les mineurs qui trouvent le nonce en premier seront payés pour l'exploitation minière.

II.14.2.4 Méthode d'exploitation

Il existe plusieurs méthodes pour l'extraction, y compris « l'extraction en solo », « l'extraction en pool » et « l'extraction en nuage ». Si vous voulez faire du minage seul, vous pouvez utiliser le "minage en solo". Cependant, votre niveau PC personnel est beaucoup moins puissant et le taux de réussite du minage est faible [88].

D'un autre côté, le "minage en pool", qui est effectué par un groupe de plusieurs personnes, signifie que chacun d'entre eux effectue le calcul de la valeur de hachage en partageant chaque plage [88].

Le "minage en nuage" est simplement une méthode pour investir dans des organisations minières et gagner des récompenses minières.

II.15 Conclusion

Dans ce chapitre, nous avons présenté dans un premier temps une introduction aux crypto-monnaies qui permettent les paiements sécurisés en se basant sur plusieurs méthodes de chiffrements. Pour cela, nous avons fait référence aux divers algorithmes de cryptage et techniques cryptographiques qui protègent ces entrées, tels que le cryptage à courbe elliptique, les paires de clés public-privé et les fonctions de hachage. Ensuite, dans un second temps, nous avons abordé le concept de cryptographie. Le but traditionnel de la cryptographie est d'élaborer des méthodes permettant d'échanger des données de manière sécurisée.

C'est pour ça la cryptographie moderne s'attaque en fait plus généralement aux problèmes de sécurité des communications. Puis dans un troisième temps, nous avons étudié le procédé technique sur lequel repose la Blockchain. Cette innovation informatique permet ainsi d'organiser les échanges de données sur un réseau distribué, assurant une sécurisation des données par chiffrement, et faisant participer les nœuds du réseau pour la création de nouveaux blocs de la chaîne. Le principe de base d'une chaîne de blocs repose sur la notion de preuve de travail, et a recours aux techniques de la cryptographie pour vérifier les détenteurs distincts d'un système d'enregistrement collectif.

Dans le chapitre suivant, nous allons présenter les réseaux informatiques.

Chapitre III : Les réseaux informatiques

III.1 Introduction

La sécurité des informations sur les réseaux informatiques reçoit de plus en plus d'attention dans la vie nationale, car de nombreuses informations importantes sont stockées sur le réseau [89]. Une fois ces informations divulguées, elles entraîneront des pertes incommensurables. La raison pour laquelle les informations du réseau sont divulguées est que, d'une part, de nombreux intrus font tout leur possible pour "voir" certaines données ou informations d'intérêt ; d'autre part, le réseau lui-même présente des risques de sécurité pour assurer le succès de l'intrus [90]. Compte tenu de ces problèmes, ce chapitre présente les connaissances de base, les méthodes techniques et les principaux développements actuels de la sécurité des réseaux liés à la sécurité des réseaux informatiques, et enseigne systématiquement l'état actuel de la sécurité des réseaux, la théorie et la technologie de la cryptographie, la communication sécurisée, les protocoles de sécurité des réseaux, les infractions et la défense de sécurité des réseaux, etc.

III.2 Généralité sur les réseaux

III.2.1 *Définition d'un réseau*

C'est une combinaison de plusieurs ordinateurs ou bien, c'est un ensemble d'équipements interconnectés par des lignes de communication, qui permet l'échange de données entre les appareils informatiques (ordinateurs, serveurs et des routeurs ...) [91].

III.2.2 *Classification des réseaux*

Les réseaux sont divisés par caractéristique territoriale, par répartition territoriale, les réseaux peuvent être locaux, mondiaux et régionaux. Les réseaux locaux sont des réseaux situés dans un ou plusieurs bâtiments. Les réseaux régionaux sont ceux situés sur le territoire d'une ville ou d'une région. Les réseaux mondiaux sont des réseaux situés sur le territoire d'un État ou d'un groupe d'États, par exemple Internet.

Dans la classification des réseaux, il existe quatre termes principaux : réseau personnel (PAN), réseau local (LAN), réseau métropolitain (MAN), et réseau géographiquement distribué (WAN) [92].

III.2.2.1 Réseau personnel PAN (Personnel Area Network)

C'est un réseau informatique organisé autour d'une personne individuelle tel que les terminaux GSM, portables, etc. interconnectés sur quelques mètres.

III.2.2.2 Les réseaux locaux LAN (Local Area Network)

« La section locale zone du réseau » relie les ordinateurs et les imprimantes qui sont généralement situés dans le même bâtiment sur de courte distance (quelques kilomètres au maximum). Chaque ordinateur connecté au réseau local est appelé poste de **travail** ou **nœud de réseau**. Ce réseau est illustré dans la figure III-*1*.

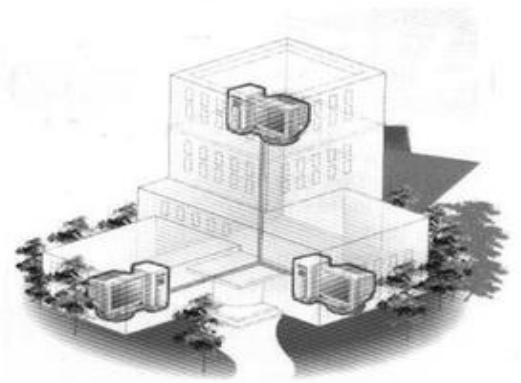


Figure III- 1: Exemple d'un Lan (Local Area Network)

On distingue les deux principaux types de LAN :

- *Architecture peer to peer:*

Cette architecture définit un modèle de réseau informatique d'égal à égal entre ordinateurs, qui distribuent et reçoivent des données ou des fichiers avec le même appareil qui agit en tant que client et en tant que serveur dans cet arrangement.

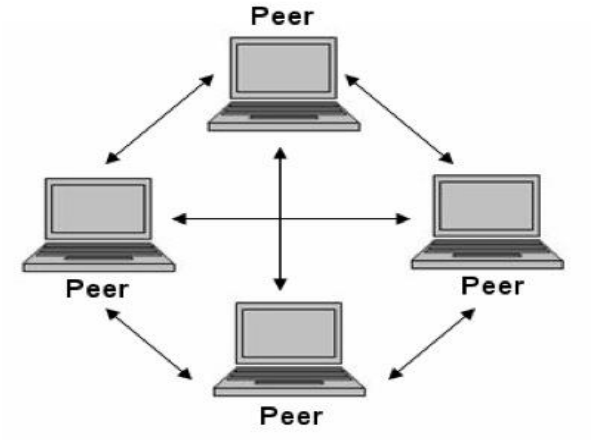


Figure III- 2: Architecture Peer to Peer

- *Architecture Client/serveur :*

Des machines clientes contactent un serveur qui leur fournit des services comme montre la figure suivante :

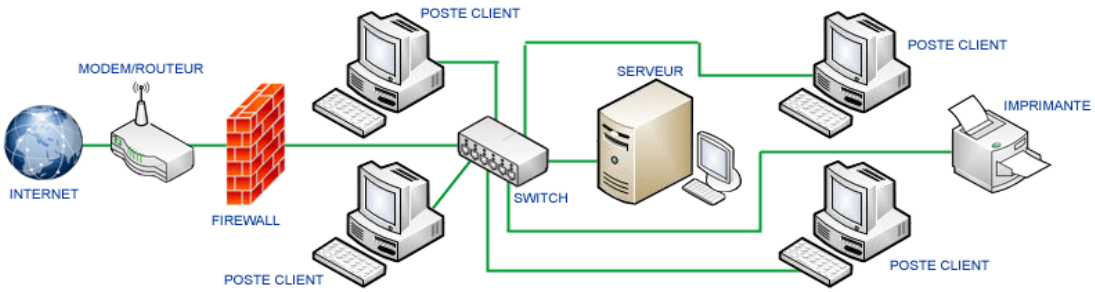


Figure III- 3: Architecture Client / Serveur

Comparaison entre architecture client/serveur et Peer to Peer architecture [92]

Architectures	Avantages	Inconvénients
Client / Serveur	Plus grande sécurité Rapidité Sauvegarde centralisée	Matériel puissant et coûteux Réseau en pane en cas d'arrêt du serveur Difficulté d'installation
Peer to Peer	Moins coûteux Pas de panne générale Facile à installer	Moins sécurisé Nombre de stations limité Ne s'adapte pas au réseau étendu

Tableau III- 1: Comparaison entre Architecture Client / Serveur et Peer to Peer

III.2.2.3 Les réseaux métropolitains MAN (Métropolitain Area Network)

C'est un *« réseau d'une grande ville »* interconnectant plusieurs réseaux LAN géographiquement proches. Ce réseau est plus petit que le réseau étendu et plus grand que le réseau local. Par exemple, une banque possédant plusieurs agences peut utiliser ce type de réseau.

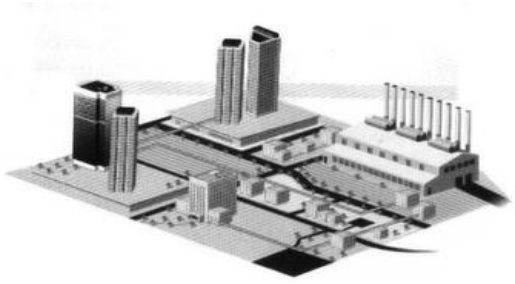


Figure III- 4: Exemple d'utilisation d'un WAN

III.2.2.4 Les réseaux étendus WAN (Wide Area Network)

C'est un réseau qui couvre une grande surface et comprend un grand nombre de nœuds. Il peut être situé dans différentes villes et pays.

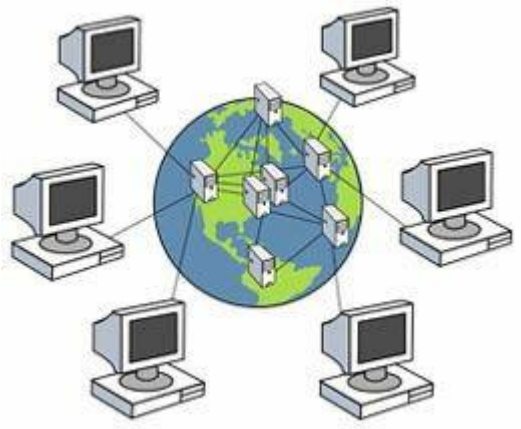


Figure III- 5: Exemple d'utilisation d'un réseau WAN

Les différentes catégories des réseaux informatiques cités auparavant sont regroupées et résumées dans les figures III-6 et III-7 :

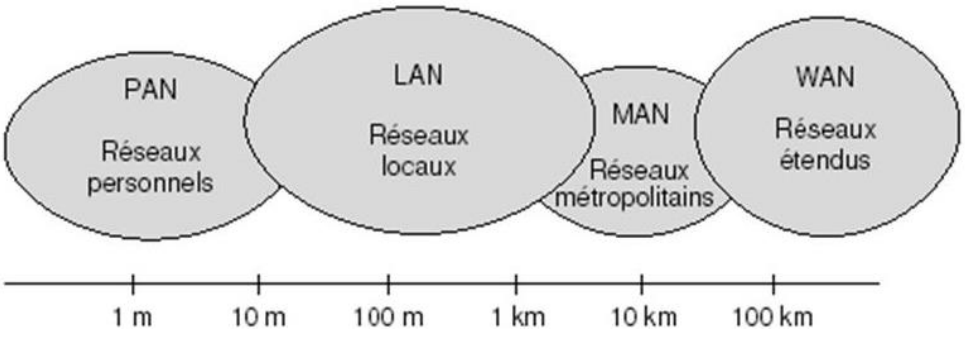


Figure III- 6: Comparaison entre le réseau selon la distance

Voilà une image qui résume tout :

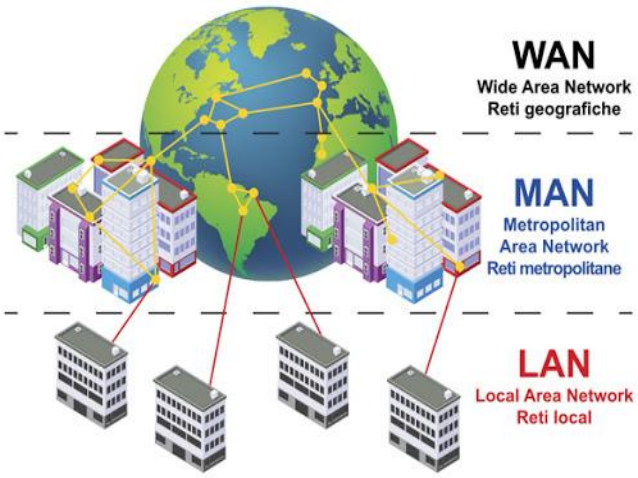


Figure III- 7: Image illustrative sur les différents types de réseau

III.3 Les types des réseaux

III.3.1 *Internet*

Internet est le système mondial de réseaux informatiques interconnectés qui utilise les protocoles Internet (TCP / IP) pour relier des appareils dans le monde entier. Il a été largement utilisé dans le monde depuis 1990 et son utilisation a été remarquable ces dernières années. Ce réseau est interconnecté par "ISP (Internet Service Provider)", un fournisseur de services qui donne accès à Internet [93].

III.3.2 *Intranet*

Un intranet est un réseau au sein d'une organisation qui utilise la technologie Internet. Ce système est conçu pour être utilisé pour le partage d'informations et le soutien aux entreprises. Intranet se base sur les mêmes technologies qu'Internet (protocole TCP/IP) [94].

III.3.3 *Extranet :*

Un extranet est un réseau sécurisé qui comprend un ensemble limité de ressources partagées. L'extranet se réfère principalement à un système de réseau qui assure la communication d'informations entre différentes entreprises telles que le commerce électronique et l'échange de données électroniques [95].

Les intranets et les extranets sont des réseaux conformes et utilisent Internet, mais ce sont toujours des réseaux fermés. Par conséquent, comme il est possible de se connecter facilement à Internet, des mesures de sécurité avancées telles que le VPN sont nécessaires pour maintenir la confidentialité.

III.4 Les Topologies des réseaux

III.4.1 Introduction

La topologie du réseau est la forme de connexion du réseau informatique. Elle indique comment les périphériques (nœuds) tels que les PC, les serveurs et les commutateurs sont connectés dans un réseau [96].

Type de topologie	*Le contenue*
Topologie physique	Configuration physique qui montre comment les câbles LAN et les PC sont réellement connectés.
Topologie logique	Une configuration logique qui représente le flux des données. Les topologies de réseau typiques incluent le type de bus, le type d'étoile, le type d'anneau et le type de maillage complet.

Tableau III- 2: Type de topologie

III.4.2 Topologie en bus

La topologie de bus connecte plusieurs nœuds à un même câble (câble coaxial). Elle fixe des résistances appelées terminateurs aux deux extrémités d'un câble coaxial pour empêcher la réflexion du signal et la perturbation à l'extrémité.

Dans une topologie de bus, si le câble coaxial central tombe en panne, tous les nœuds connectés ne pourront pas communiquer [96].

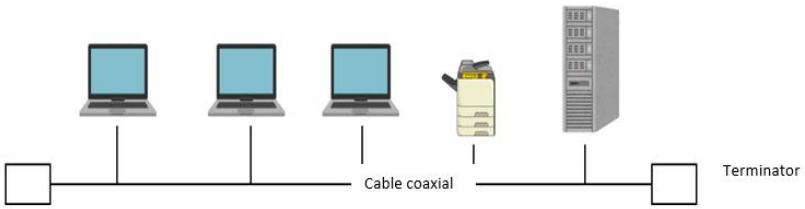


Figure III- 8: Topologie en bus

III.4.3 Topologie en étoile

Une topologie en étoile relie plusieurs nœuds à un concentrateur (concentrateur ou commutateur). Contrairement à une topologie de bus, une défaillance d'un câble n'affecte pas la communication des autres nœuds, mais une défaillance du concentrateur affecte tous les nœuds. Aujourd'hui, ce type de LAN est la topologie dominante [96].

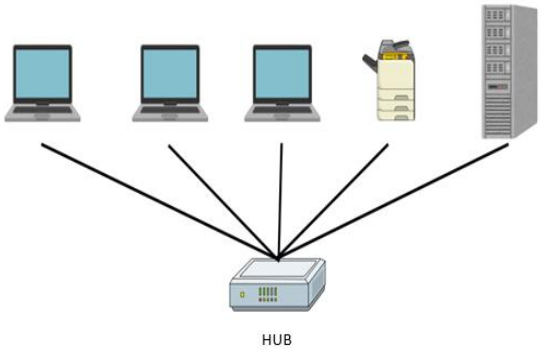


Figure III- 9: Topologie en étoile

III.4.4 Topologie en anneau

Dans une topologie en anneau, les nœuds sont logiquement connectés dans un anneau et les données appelées jetons circulent à grande vitesse autour de l'anneau. Le nœud qui obtient ce jeton envoie les données. Dans une topologie à anneau unique, en cas de défaillance unique, l'ensemble du réseau ne pourra pas communiquer. Une topologie à anneau unique avec une tolérance aux pannes améliorée devient une topologie à anneau double.

Dans cette topologie, en fournissant un autre anneau de rechange, la communication peut être poursuivie même en cas de défaillance de l'anneau principal [96].

Topologie en anneau unique ***Topologie à double anneau***

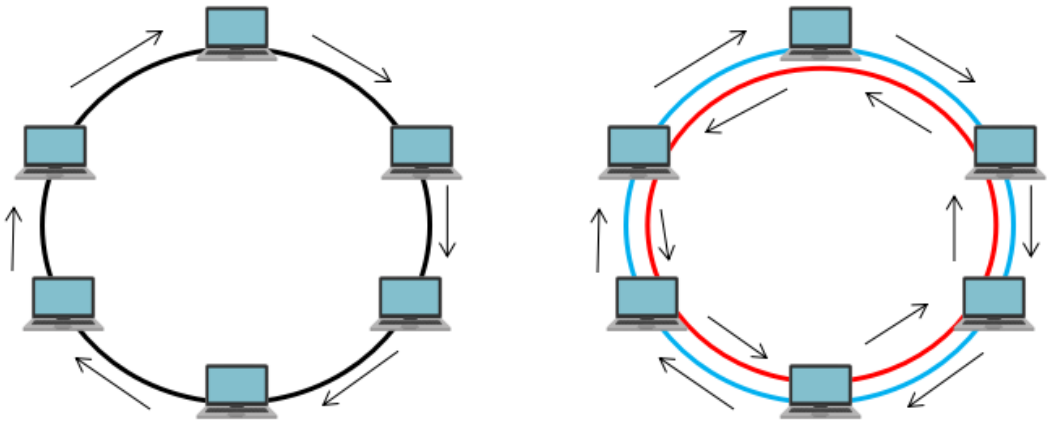


Figure III- 10: Topologie en anneau

III.4.5 Topologie maillée complète

Dans le cas du type à maillage complet, chaque nœud est interconnecté avec tous les routeurs, et même si une liaison avec un routeur spécifique échoue, la communication peut être poursuivie via la liaison avec un autre routeur.

Bien qu'une configuration à maillage complet soit hautement tolérante aux pannes, l'utilisation d'un maillage complet dans une topologie physique augmente les coûts. Par conséquent, l'utilisation d'un maillage partiel, dans lequel seul le routeur central et

certains des routeurs sont maillés, peut réduire les coûts et maintenir une certaine redondance et tolérance aux pannes [96].

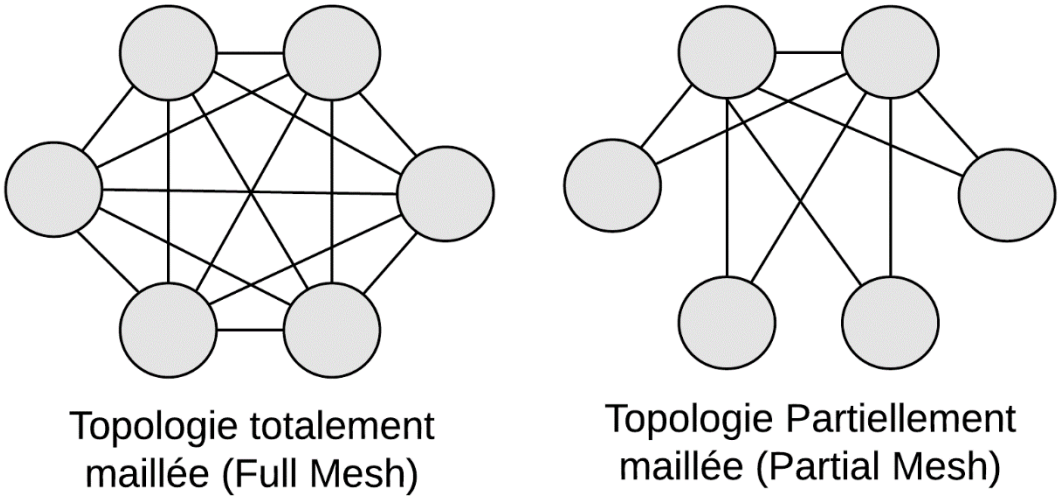


Figure III- 11: Topologie maillé

III.5 Equipements d'interconnexion

III.5.1 Répéteur

Un répéteur est un périphérique qui connecte des réseaux au niveau de la couche physique, qui est la première couche du modèle de référence de base OSI. Les répéteurs peuvent étendre la distance de transmission en amplifiant le signal [97].

III.5.2 Pont

Un pont (bridge) est un périphérique qui effectue une connexion réseau **dans la couche liaison de données**, **qui est la deuxième couche** du modèle de référence de base OSI. Il détermine l'**adresse MAC**, qui est une *valeur unique* attribuée à une carte d'interface réseau (NIC[28]), et les données de relais [97]. Il permettant de relier des réseaux de même nature.

III.5.3 Routeur

Le routeur est un périphérique de connexion qui connecte le réseau local et le réseau étendu. Il s'agit du périphérique central du réseau. Lorsque des données sont transmises d'un sous-réseau à un autre, cela se fait via un routeur. Le routeur choisit le meilleur chemin en fonction du coût de transmission, du délai de transit, de l'encombrement du réseau ou de la distance entre la source et la destination [97].

[28] **NIC** : Une carte réseau est un composant qui fournit des capacités de mise en réseau pour un ordinateur. Il peut permettre une connexion filaire (telle qu'Ethernet) ou une connexion sans fil (telle que Wi-Fi) à un réseau local.

III.5.4 Passerelle

La passerelle est un appareil qui connecte deux réseaux informatiques avec différents protocoles réseau et différentes architectures. Il existe deux types de passerelles : l'une est une passerelle orientée connexion et l'autre est une passerelle sans connexion [97].

III.5.5 Concentrateur

Un concentrateur (Hub) est un répéteur régénérateur de signaux qui peut distribuer des signaux sur plusieurs lignes. Une extrémité du concentrateur a une interface avec le serveur et l'autre extrémité à plusieurs interfaces avec la station de travail réseau. Le nombre d'interfaces de concentrateur détermine le nombre d'ordinateurs connectés au réseau [97].

III.5.6 Commutateur

Un commutateur (Switch) est un dispositif permettant de relier divers éléments tout en segmentant le réseau [97].

III.5.7 Adaptateur

Un adaptateur (adapter) est un dispositif permettant de connecter deux systèmes qui n'avaient pas été conçus pour cela à l'origine [97].

III.6 Le modèle de référence OSI

III.6.1 Présentation du modèle OSI

Le but initial du réseau était de connecter des ordinateurs et des appareils, mais comme chaque entreprise a développé son propre réseau, il est devenu impossible de se connecter facilement au réseau, de sorte qu'un modèle unifié du réseau a été créé pour la communauté. Ce modèle est appelé modèle OSI.

Le protocole OSI est une norme établie par l'organisation internationale de normalisation (ISO). Il s'agit d'un modèle simplifié qui divise les fonctions requises pour la communication en sept couches. C'est ce qu'on appelle le "modèle de référence OSI". Il sert également d'indicateur de la conception actuelle du réseau et est utilisé pour représenter l'équipement réseau [98].

III.6.2 Les couches du modèle OSI

Couche Physique : Elle définit le contenu physique requis pour transmettre un signal. Plus précisément, elle définit les niveaux de tension, les synchronisations de commutation, les taux de transmission, les distances de transmission, les formes des connecteurs, etc [98].

Couche de liaison de données : La couche liaison de données définit les protocoles requis pour relier les couches physique et réseau (L'adresse physique, la vérification de la trame de données, la gestion du flux de transmission / réception, la topologie prise en charge, etc.). Elle est divisée en deux sous-couches [98] :

- La couche MAC qui structure les bits de données en trames.
- La couche LLC qui assure le transport des trames.

Couche réseau : La couche 2 permet la communication entre les terminaux connectés au même commutateur ou concentrateur. La portée de ce réseau est un LAN (réseau local). La couche 3 fournit des connexions LAN à LAN. En d'autres termes, "l'interfonctionnement" qui interconnecte les réseaux.

Couche de transport : Cette couche est utilisée pour corriger les erreurs et absorber les différences de taille des blocs de données (telles que la division de grandes données en paquets plus petits).

Couche de session : Elle spécifie la procédure de connexion entre des programmes tels que le client et le serveur. Cette couche fournit un canal de communication logique pour l'échange de données entre les deux programmes.

Couche de présentation : Elle Gère les formats de représentation des données, tels que les types de codes de caractères et le chiffrement. Des traitements tels que la conversion lorsque les codes de caractères sont différents entre les deux appareils et le chiffrement et le déchiffrement des communications sont effectués dans cette couche.

Couche d'application : Elle est définie comme une interface utilisateur. Elle gère des fonctions telles que l'identification des destinations de communication et la synchronisation des communications.

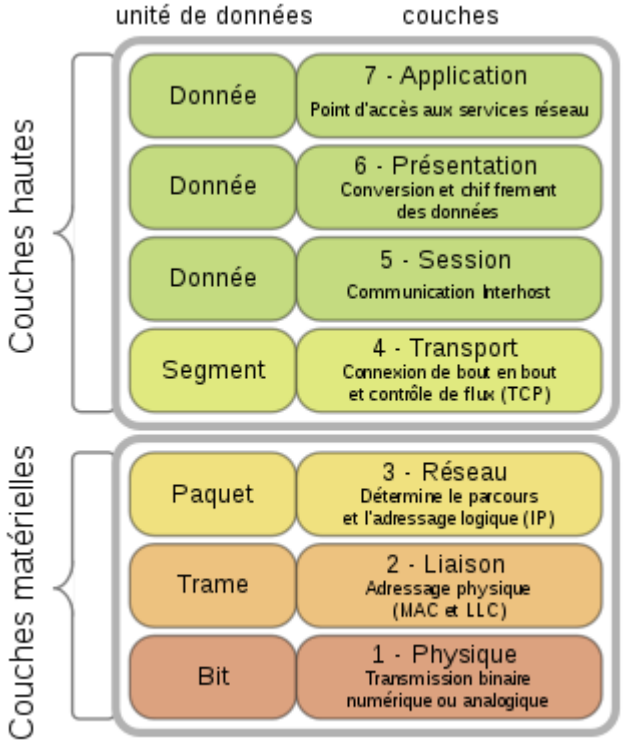


Figure III- 12: Modèle OSI

III.7 Le modèle TCP/IP

III.7.1 Présentation du modèle TCP/IP

TCP / IP est l'abréviation de « Transmission Control Protocol / Internet Protocol ». C'est la norme la plus utilisée au monde et est devenue la norme de facto. Les fonctions requises

pour la communication sont divisées en quatre couches et l'architecture du réseau est plus simple que le modèle de référence OSI. La fonctionnalité de chaque couche peut être appliquée au modèle de référence OSI. Tous les appareils et logiciels utilisés sur Internet sont conformes à TCP/IP [99].

III.7.2 La différence entre le modèle de référence OSI et le modèle de couche TCP / IP

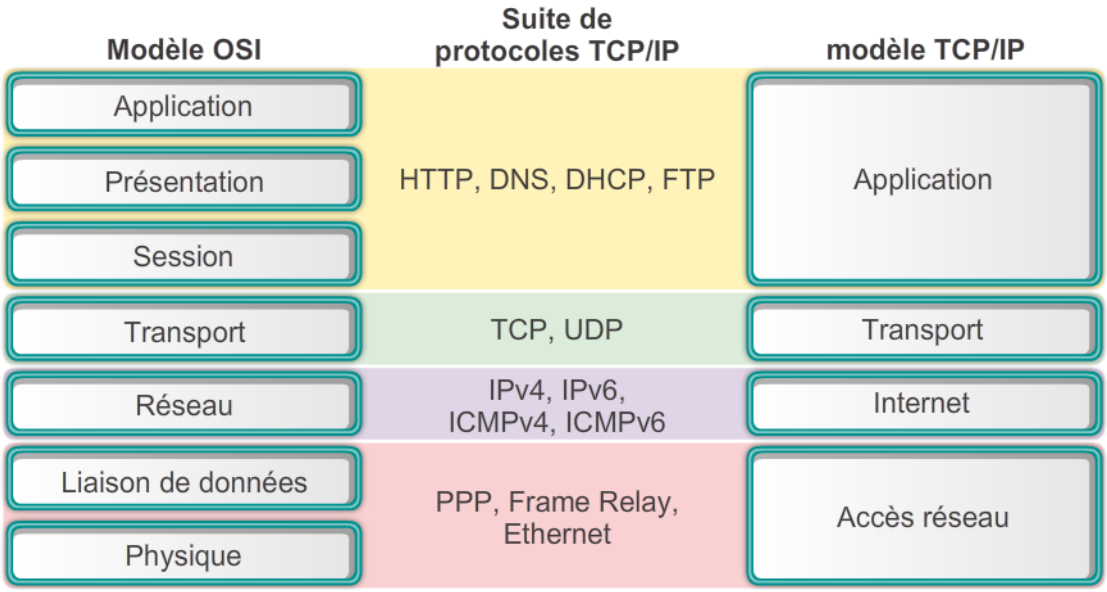


Figure III- 13: Déférence entre OSI et TCP/IP

III.8 L'adressage IP

III.8.1 Qu'est-ce qu'une adresse IP ?

L'adresse IP (Internet Protocol Address) est le numéro de l'appareil connecté à Internet. Lors de l'échange de données, elle est utilisée pour vous assurer de ne pas confondre le partenaire de communication sur le réseau. Il existe des types et des règles pour les adresses IP [100].

III.8.2 Format des adresses IP

Chaque adresse IP comporte deux parties : un ID de réseau, et un ID d'hôte.

Tous les hôtes d'un même réseau doivent avoir le même ID réseau, unique dans l'inter-réseau. L'ID d'hôte identifie une station de travail, un serveur, un routeur ou tout autre hôte TCP/IP du réseau. L'ID d'hôte doit être unique pour chaque ID de réseau.

Deux formats permettent de faire référence à une adresse IP : le format binaire et la notation décimale pointée. Chaque adresse IP a une longueur de 32 bits et est composée de quatre champs de 8 bits, qualifiés d'octets. Les octets sont séparés par des points et représentent un nombre décimal compris entre 0 et 255. Les 32 bits de l'adresse IP sont alloués à l'ID de réseau et à l'ID d'hôte [101].

III.8.2.1 Adresse d'hôte et adresse de réseau

Une adresse IPv4 est une suite de 32 bits (4 octets) exprimée en décimale à point, en séparant chacun des octets par un point.

Prenons l'exemple ci-dessous :

Représentation binaire : **1100000000. 10101000. 0s0000101. 00000000**

$$1 * 2^7 + 1 * 2^6 + 0 * 2^5 + 0 * 2^4 + 0 * 2^3 + 0 * 2^2 + 0 * 2^1 + 0 * 2^0$$

Bits de poids fort Bits de poids faible

Représentation décimale pointée : **192. 168. 5. 0**

Dans ce chapitre, on vous a présenté l'adressage IP de base. Vous connaissez tous les 3 classes principales d'adresses et des adresses spécifiques IP. Mais, il vous manque encore deux notions impératives dans l'adressage IP.

Il est essentiel de comprendre comment définir un plan d'adressage et quels sont les pièges !

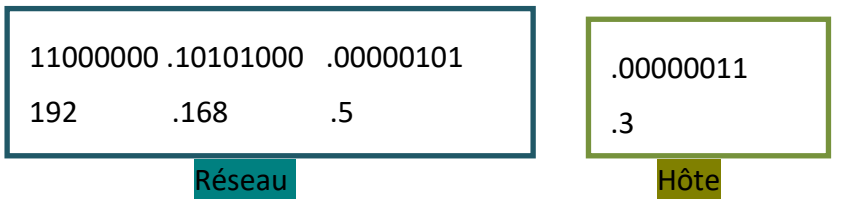
Comment définir un plan d'adressage satisfaisant ?

III.8.2.2 Structure d'une adresse

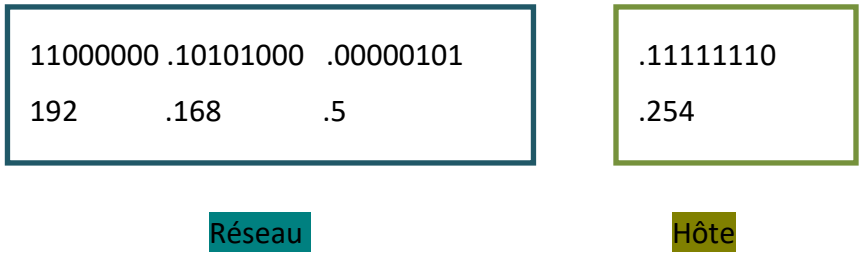
Les premiers hôtes du réseau :

11000000 .10101000 .00000101	.00000001
192 .168 .5	.1
Réseau	Hôte

11000000 .10101000 .00000101	.00000010
192 .168 .5	.2
Réseau	Hôte



Le dernier hôte du réseau :



Sur la base de ce qui précède, nous entrerons dans le sujet principal. L'adresse IP contient les informations "Quel réseau, quel ordinateur ?"

La partie « De quel réseau s'agit-il ?» Est appelée « partie réseau ».

La partie "De quel ordinateur s'agit-il ?" Est la " partie hôte ". Ainsi, d'où à où dans l'adresse IP est "quel est le réseau ? " Information, et d'où à où est "quel ordinateur est-il ? "

Ceci est indiqué par le « masque net ».

III.8.2.3 Le masque

Chaque hôte d'un réseau nécessite un masque de sous-réseau avec une adresse 32 bits utilisée pour bloquer une partie de l'adresse IP afin de distinguer l'ID de réseau à partir de l'ID d'hôte. Donc, c'est une série continue de 1 (partie gauche) et la partie qui correspond aux hôtes est une série continue de 0 (partie droite). Le masque est aussi exprimé en notation décimale pointée.

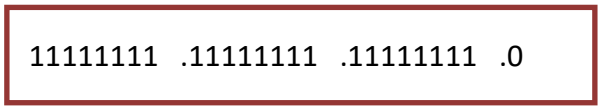
En comparant le masque de réseau et l'adresse IP, vous pouvez voir où se trouve la partie réseau et où se trouve l'hôte.

Par exemple, supposons que l'adresse IP soit « 192.168.5.2 » et le masque de réseau « 255.255.255.0»

Adresse IP : 192. 168. 5. 2

Masque : 255. 255. 255.0

Le masque est aussi exprimé en notation binaire est la suivante :



Mettons ces deux côtes à côte c'est-à-dire l'adresse IP et le masque. On fait Une addition logique entre l'adresse IP d'un poste et son masque permet de déterminer l'adresse du réseau à laquelle appartient le poste.

III.8.3.1 MASQUES DE SOUS-RÉSEAUX PAR DÉFAUT

Nous savons tous que l'IP est composé de quatre segments de nombres. Ici, nous examinons d'abord les trois types de classe, comme illustre le tableau suivant [101]:

Classe	**Bits de départ**	**Début**	**Fin**	***Masque de sous-réseau par défaut***
Classe A	0	0.0.0.0	128.255.255.255	255.0.0.0
Classe B	10	128.0.0.0	191.255.255.255	255.255.0.0
Classe C	110	192.0.0.0	223.255.255.255	255.255.255.0

Tableau III- 3: Résumé l'adressage de masque des classes

III.8.2.4 Calcul d'adresse réseau

Elle est calculée par application en binaire du masque sur l'adresse IP en utilisant la fonction ET (AND logique).

Entrer		**Sortie**
A	B	A ET B
1	0	0
0	1	0
0	0	0
1	1	1

Tableau III- 4: Table de vérité

Exemple de Classe C : @IP 192.168.5.3 avec le masque 255.255.255.0, Calculer l'adresse du réseau ?

@ip hôte 11000000 .10101000 .00000101 .00000011

ET

@masque 11111111 .11111111 .11111111 .00000000

@ip de réseau 11000000 .10101000 .00000101 .0000000

III.8.2.4 Calcul de l'adresse de diffusion

Chaque réseau possède une adresse particulière dite de diffusion. Tous les paquets avec cette adresse de destination sont traités par tous les hôtes du réseau local. L'adresse de diffusion du réseau est la dernière adresse du réseau.

11000000 .10101000 .00000101 .11111111

24 bits pour le réseau

8 bits à 1 pour la partie hôte

Elle est donc constituée en positionnant tous les bits de l'hôte à 1.

III.8.2.5 Calcul de la plage adressable

La plage adressable est l'ensemble des adresses que peut prendre un hôte sur le réseau.

192 .168 .5 11000000 .10101000 .00000101	**.1** **.00000001**

La première adresse de la plage

192 .168 .5 11000000 .10101000 .00000101	**.254** **.11111110**

La dernière adresse de la plage

III.8.2.6 Nombre d'hôtes possibles dans un réseau

192 .168 .5 1100000000. 10101000. 00000101	.0 .00000000

Le nombre de bits contenus dans la partie hôte détermine le nombre d'hôtes possible sur un réseau ==> 8 bits pour les hôtes ==>$2^8 - 2$ hôtes possibles.

Pourquoi 2 ?

Parce que deux adresses sont réservées et ne peuvent être affectées à un hôte :

- La première adresse (192.168.5.0) représente l'adresse du réseau.
- La dernière adresse (192.168.5.255) représente l'adresse de diffusion du réseau.

III.8.2.7 La notation CIDR du masque

Le CIDR est principalement une norme basée sur le préfixe au niveau du bit pour l'interprétation des adresses IP. Elle le fait en une combinaison d'une pluralité de blocs d'adresse à une table d'acheminement. Ces blocs d'adresses sont appelés blocs d'adresses CIDR.

Le masque est constitué d'une suite contiguë de 1 suivie d'une suite de 0 ; l'information utile est le nombre de 1 dans le masque.

Une autre notation (la plus utilisée actuellement) consiste à faire suivre une adresse donnée par le nombre de bits égaux à 1 dans le masque.

Par exemple : 192.168.5.0 avec le masque 255.255.255.0 correspond à 192.168.5.0/24

Le tableau III-5 illustre l'adressage IP [101].

Classe	Bits de départ	Début	Fin	Notation CIDR par défaut	Masque de sous-réseau par défaut
Classe A	0	0.0.0.0	128.255.255.255	/8	255.0.0.0
Classe B	10	128.0.0.0	191.255.255.255	/16	255.255.0.0
Classe C	110	192.0.0.0	223.255.255.255	/24	255.255.255.0
Classe D (multicast)	1110	224.0.0.0	239.255.255.255		255.255.255.255
Classe E (réservée)	1111	240.0.0.0	255.255.255.255		Non défini

Tableau III- 5: Tableau illustratif pour l'adressage IP

III.9 Les protocoles de routage

III.9.1 Introduction

Dans le monde des réseaux IP, le rôle de "routage" qui relie les paquets est très important. Si vous allez maîtriser le domaine des réseaux IP, il est essentiel de comprendre le fonctionnement du routage.

Il existe le « routage statique » et le « routage dynamique » comme méthodes de gestion des informations de routage dans les réseaux IP. Le routage statique est une méthode de définition manuelle des informations de routage dans chaque routeur. Ces informations de routage ne disparaissent pas fondamentalement de la table de routage.

Comparons brièvement les forces et les faiblesses de ces deux approches.

L'avantage du routage statique est qu'il peut fournir une accessibilité stable au réseau car les informations de route gérée ne sont pas fondamentalement supprimées de la table de routage. De plus, comme les informations de routage sont définies manuellement, aucun traitement CPU ni trafic n'est requis pour l'échange d'informations. Cependant, même lorsque le réseau de destination n'existe plus, le trafic est transféré sur la base de ces informations.

L'avantage du routage dynamique est que les informations de routage sont apprises dynamiquement par le protocole de routage, éliminant ainsi le besoin de gestion. De plus, comme les mises à jour du réseau peuvent être reflétées dynamiquement, il est possible de rejeter rapidement le trafic destiné aux routes non disponibles ou de sélectionner un circuit approprié. Cependant, si des informations de route incorrectes sont annoncées en raison d'une erreur de configuration ou d'une panne de périphérique, ces informations sont également transmises à l'ensemble du réseau, ce qui entraîne un large éventail d'échecs de communication. Dans le pire des cas, une boucle de routage peut se produire, entraînant

une augmentation rapide de la charge du processeur et du trafic, et le réseau lui-même peut tomber en panne.

Lorsque vous décidez de la méthode à utiliser lors de la création d'un réseau, vous devez tenir compte des forces et des faiblesses des deux méthodes. En pratique, cependant, il est rare de construire un réseau sans aucun routage dynamique. Avec la propagation d'Internet et la taille des réseaux IP de plus en plus importants, il est pratiquement impossible de gérer les informations de routage uniquement manuellement dans un réseau aussi vaste. De plus, comme diverses applications s'exécutent sur IP, il y aura naturellement des demandes de redondance réseau et de contrôle du trafic. Pour répondre à ces demandes, le routage dynamique reste indispensable [102].

III.9.2 Protocoles de routage qui composent Internet

Les protocoles de routage peuvent être largement classés en deux types : IGP (Interior Gateway Protocol) et EGP (Exterior Gateway Protocol). Les protocoles utilisés comme IGP incluent "RIP", "OSPF", "IS-IS" et "IGRP / EIGRP (protocole propriétaire de Cisco Systems)". RIP et IGRP conviennent aux petits réseaux. OSPF, IS-IS, EIGRP, etc. conviennent mieux aux réseaux moyens et plus importants. Plus précisément, les réseaux d'entreprise utilisent souvent RIP, OSPF, EIGRP, etc., et les réseaux tels que les opérateurs et les fournisseurs de services utilisent souvent OSPF ou IS-IS. EGP a été utilisé lorsque l'Internet a commencé à émerger, mais ce protocole de routage est désormais rarement utilisé. Cela est dû au fait que malgré l'utilisation répandue d'EGP, il y avait plusieurs problèmes, tels que le traitement des boucles de routage et les restrictions de topologie [102].

III.9.3 Protocoles de routage clés et leurs combinaisons

Ici, nous nous concentrerons sur deux protocoles de routage principaux RIP et OSPF, en présentant leurs fonctionnalités [103].

III.9.3.1 RIP (Routing Information Protocol)

Le RIP n'est pas adapté à une utilisation dans de grands réseaux complexes, mais en raison de sa simplicité, il est utilisé par de nombreuses entreprises.

- Les routeurs voisins échangent des messages de mise à jour toutes les 30 secondes et mettent à jour la table de routage.
- S'il existe plusieurs itinéraires, l'itinéraire qui minimise le nombre de routeurs qui passent est sélectionné.
- Si le nombre de routeurs dépasse 15, il est considéré comme inaccessible.
- Si aucun message de mise à jour n'est reçu d'un routeur pendant 180 secondes, il est supposé qu'une défaillance s'est produite dans le routeur ou le réseau connecté, et l'itinéraire correspondant est invalidé.
- Si un itinéraire invalidé passe encore 120 secondes, il sera supprimé de la table de routage.

III.9.3.2 OSPF (protocol Open Shortest Path First)

Compte tenu de la vitesse de communication du réseau passant, qui n'est pas prise en compte dans le RIP, une sélection de route plus efficace peut être effectuée. Dans OSPF, les

informations sont collectées auprès de tous les routeurs et, sur la base des informations, la configuration du réseau telle que la contiguïté du routeur est saisie et une table de routage est créée. OSPF présente les caractéristiques suivantes par rapport à RIP et convient aux réseaux à grande échelle [104]:

- Moins susceptible de provoquer une boucle de routage (indiquer où les boucles de routes sélectionnées)
- Convergence rapide (tous les routeurs finissent de mettre à jour la table de routage).
- Les itinéraires sont sélectionnés en fonction de la bande passante et non du nombre de routeurs.

Voici un tableau comparant les fonctionnalités de RIP et OSPF [105]:

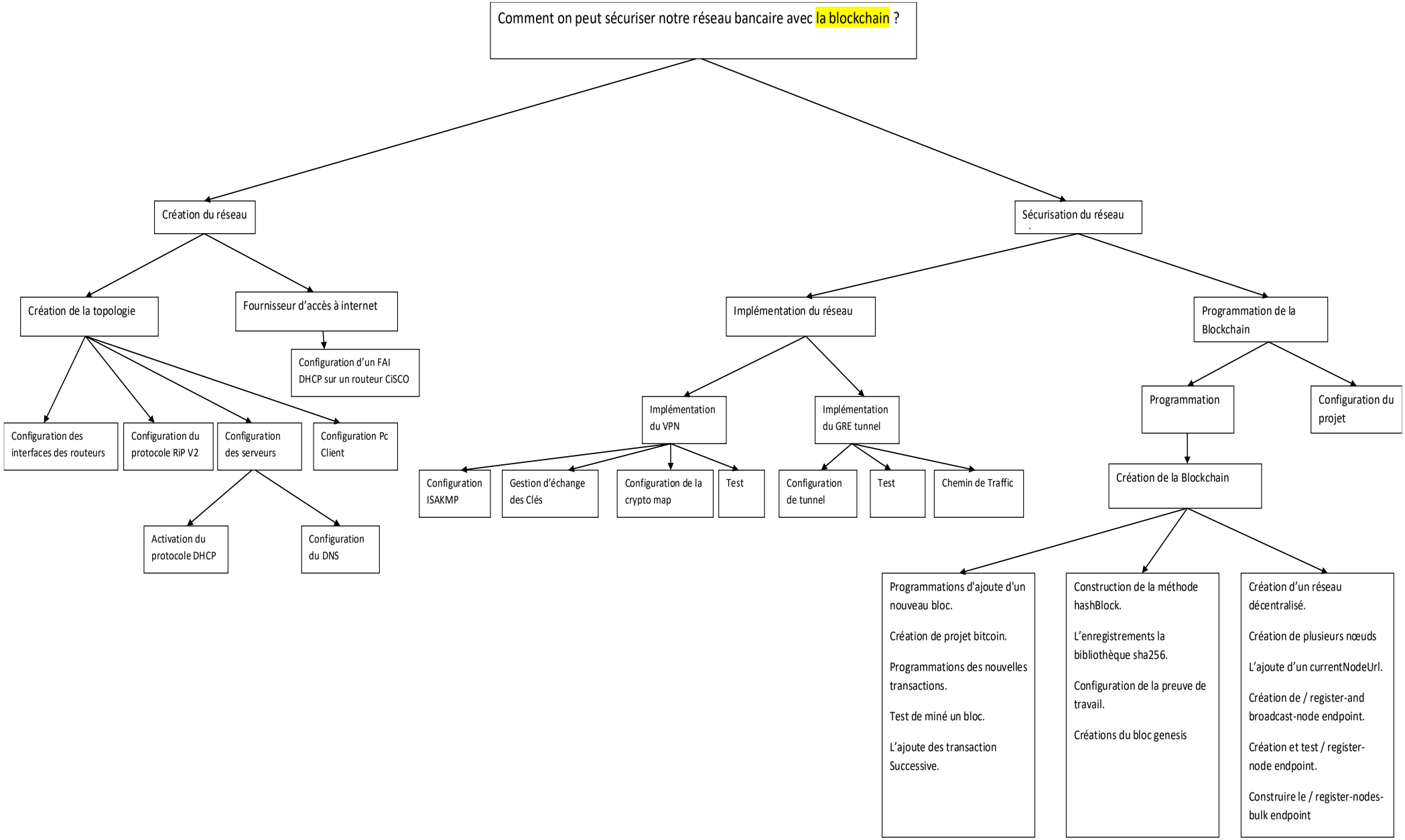
	Transmission d'informations sur l'itinéraire	**Vitesse de convergence**	**Itinéraire optimal**	**Équilibrage de charge**	**VLSM**	**Authentification**	**Difficulté de montage**
RIP1	Régulier (tous les itinéraires)	Lent	Nombre de sauts	Aucun	Non pris en charge	Non pris en charge	Faible
RIP2	Régulier (tous les itinéraires)	Lent	Nombre de sauts	Aucun	Correspondance	Correspondance	Faible
OSPF	Lors du changement (seule différence)	Rapide	Valeur du coût	Oui	Correspondance	Correspondance	Élevé

Tableau III- 6: Fonctionnalités de RIP et OSPF

III.10 Conclusion

Dans ce chapitre, on a défini les notions fondamentales dans les réseaux informatiques et présenté les différents modèles des réseaux, ainsi vous verrez pourquoi l'adressage IP est à la base des réseaux informatiques. Nous avons abordé aussi l'adressage IPv4, le découpage d'adresse IP en masque de sous réseau ainsi que les différents types d'adresse. Nous avons aussi présenté une vue globale sur le modèle OSI et TCP/IP, ainsi les notions de protocole et routage.

Chapitre IV : Sécuriser un réseau bancaire avec la blockchain



Logiciels utilisés : Node.js – Sublime – cmd – Cisco Packet Tracer.

Réaliser par :

- Mlle. BOUDGHENE STAMBOULI Kawter.
- Mlle. BOUDJEMAA Chaimaa.

Sous la supervision de :

- Mr. HADJILA Mourad.
- Mr. BENMOUSSAT ChemsEddin.

IV.1 Introduction

Comme nous l'avons vu dans les chapitres précédents, la technologie blockchain consiste une évolution informatique dans le domaine de la sécurité et la disputation des données. Cette technologie donne un support robuste et sécurisé pour partager la base de données qui enregistre toutes les transactions ou événements numériques qui ont été exécutés et partagés entre les parties participantes. Chaque transaction est vérifiée par la majorité des participants du système. Il contient chaque enregistrement unique de chaque transaction.

Actuellement, la crypto-monnaie constitue l'application la plus connue qui a efficacement exploité la manière dont une blockchain garantit l'authentification, la non-répudiation et la confidentialité des données sans avoir recours à des tiers de confiance tels que les banques ou bien les états. Cependant, la blockchain enregistre la transaction dans un grand livre numérique distribué sur le réseau, ce qui la rend incorruptible. Tout élément de valeur comme les actifs fonciers, les voitures, etc. peut être enregistré sur Blockchain en tant que transaction.

Dans ce chapitre, on va proposer l'exploitation de la blockchain pour faire des transactions sur un réseau bancaire.

IV.2 Présentation de projet

IV.2.1 *La problématique*

Chaque année, les banques perdent entre 15 et 25 milliards de dollars américains, engloutis dans des fraudes à l'identité. Malgré tout, les banques algériennes doivent intégrer cette technologie pour lutter contre le blanchiment.

« Pour l'intégration ou l'ouverture de compte, cette technologie permet aux clients d'utiliser une empreinte numérique qui, à l'instar d'une empreinte réelle, peut être utilisée comme identifiant unique ».

Notre objectif sera donc de mettre fin à ces fraudes par le bais d'une blockchain. Donc, le problème qui se pose c'est « comment peut-on sécuriser notre réseau bancaire avec la blockchain ? »

IV.2.2 *Objectif*

L'objectif dans ce projet est de créer un système de gestion et de transfert de monnaie à base de la technologie Blockchain, qui est sans doute le moyen le plus efficace, sécurisé et le plus transparent, pour gérer le transfert de la monnaie.

La blockchain pourrait offrir des nouvelles solutions pour lutter contre la corruption, puisqu'elle permet de créer et de stocker des enregistrements chiffrés qui peuvent être vérifiés, mais ne peuvent pas être modifiés ou supprimés.

Pour ces raisons, on propose de déployer une blockchain pour gérer toutes les transactions de la monnaie au niveau national.

Pour crée un réseau bancaire, il faut diviser notre travail en trois parties. La première consiste de créer un réseau bancaire à l'aide d'un logiciel qui s'appelle Cisco Packet Tracer, la

deuxième partie programme la blockchain et la troisième partie consiste à intégrer le programme exécutable sur notre réseau bancaire.

Partie 1 : Construction du réseau

IV.3 Introduction

Dans ce chapitre, nous commencerons par une présentation globale du réseau, nous expliquerons comment se fait le routage et la commutation entre les zones et les différents blocs de chaque zone, ensuite nous essayerons de voir les points faibles du réseau et donnerons des suggestions afin d'améliorer la sécurité et de rendre le trafic plus efficace. Enfin, nous présenterons l'architecture améliorée du réseau.

IV.4 Présentation globale du réseau intranet

Le réseau informatique d'un réseau bancaire est constitué de quatre zones. La figure IV.1 représente sa topologie physique.

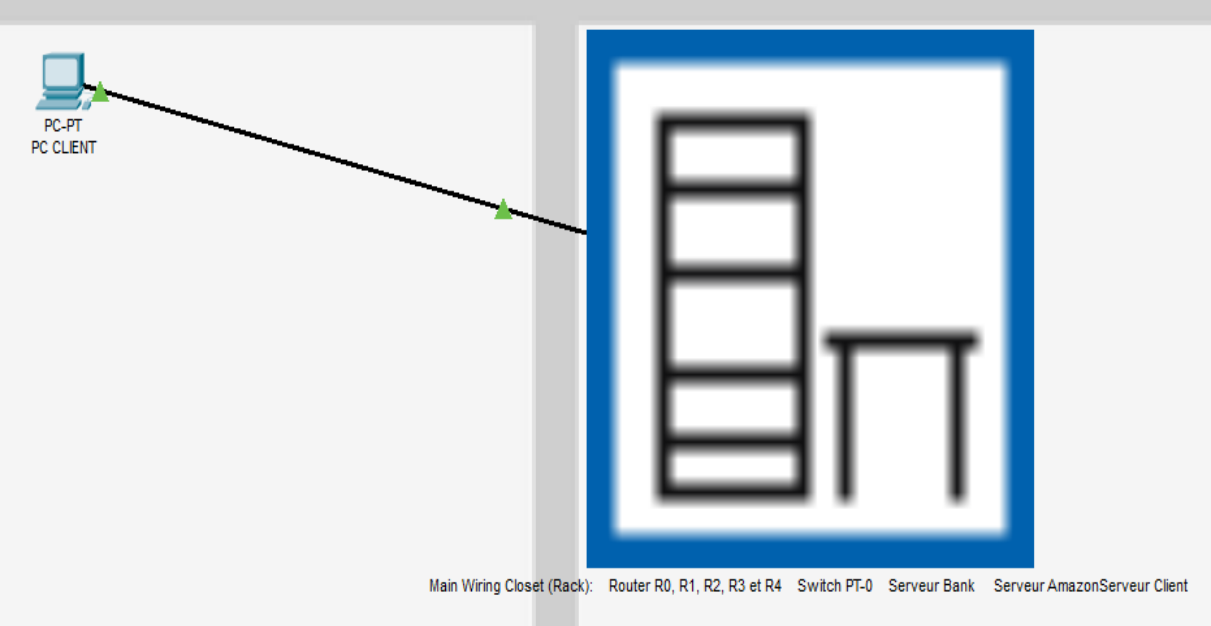


Figure IV- 1: Topologie physique du réseau

IV.4.1 Description détaillée du réseau

Nous allons réaliser, sur Packet Tracer, le schéma ci-dessous ayant cinq (5) routeurs Cisco interconnectés en port série (serial) et Ethernets.

Ce type de liaisons série nécessite une horloge de référence (clock rate) pour synchroniser la transmission des données. Aujourd'hui, ce type de liaison est obsolète mais il est toujours présent dans des anciennes topologies réseau. Il est remplacé par des liaisons Ethernet.

Nous avons constaté le choix des réseaux en adéquation avec les routeurs nous permettant de bien nous situer dans le travail. Par exemple, Nous avons choisi le réseau

10.0.0.0/8 pour interconnecter le serveur Bank et le routeur R3. Pour le réseau d'amazone, nous avons préféré d'utiliser l'adresse d'amazone.fr 45.0.0.0/8

IV.4.2 Création de la topologie du réseau

Cette étape consiste à faire communiquer les différents postes des réseaux et tester le routage appliqué entre différents réseaux qui est le routage dynamique et faire l'analyse de fonctionnement de ce protocole.

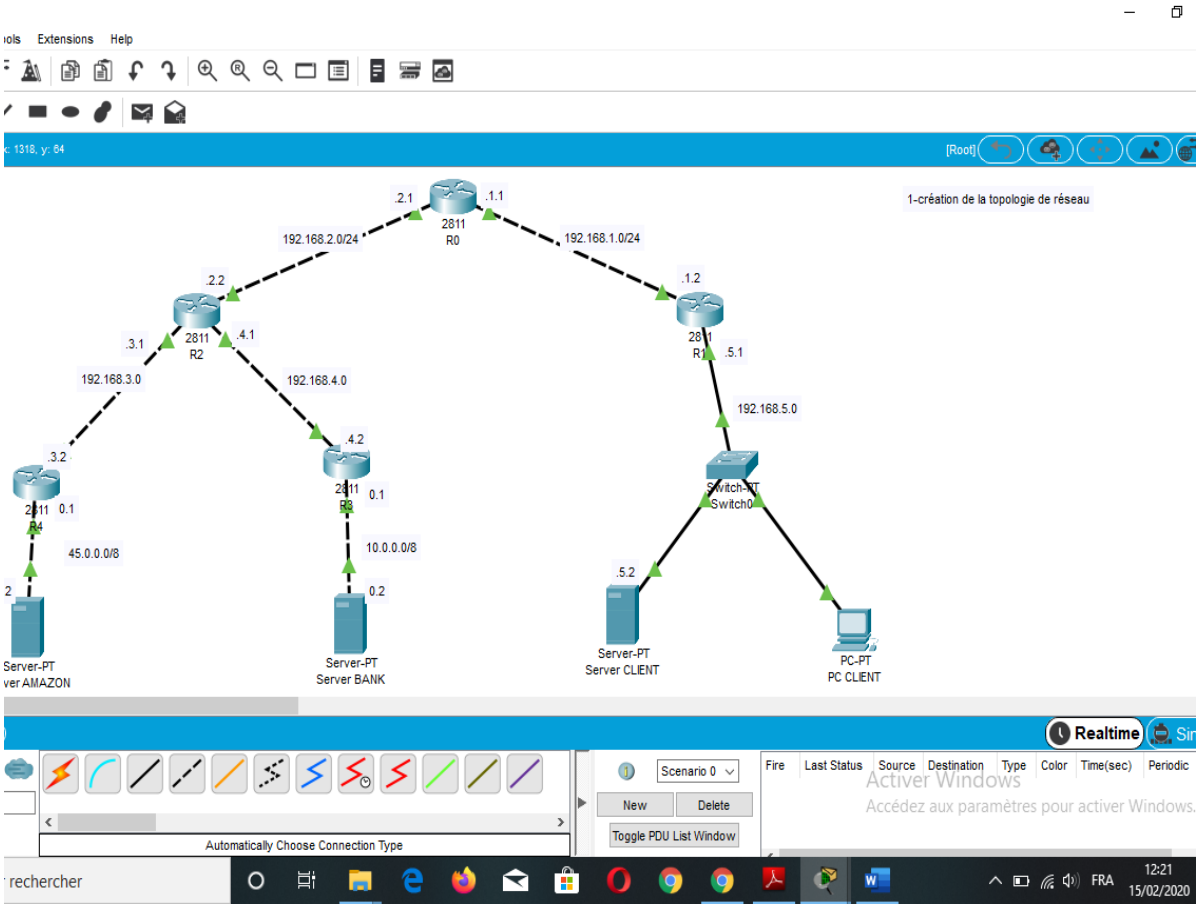


Figure IV- 2: Architecture du réseau

Nous allons maintenant configurer les 5 routeurs selon le schéma. Nous commençons par configurer les adresses ip des interfaces, ensuite configurer le routage rip.

IV.4.2.1 Configuration des interfaces des routeurs

Router R2 :

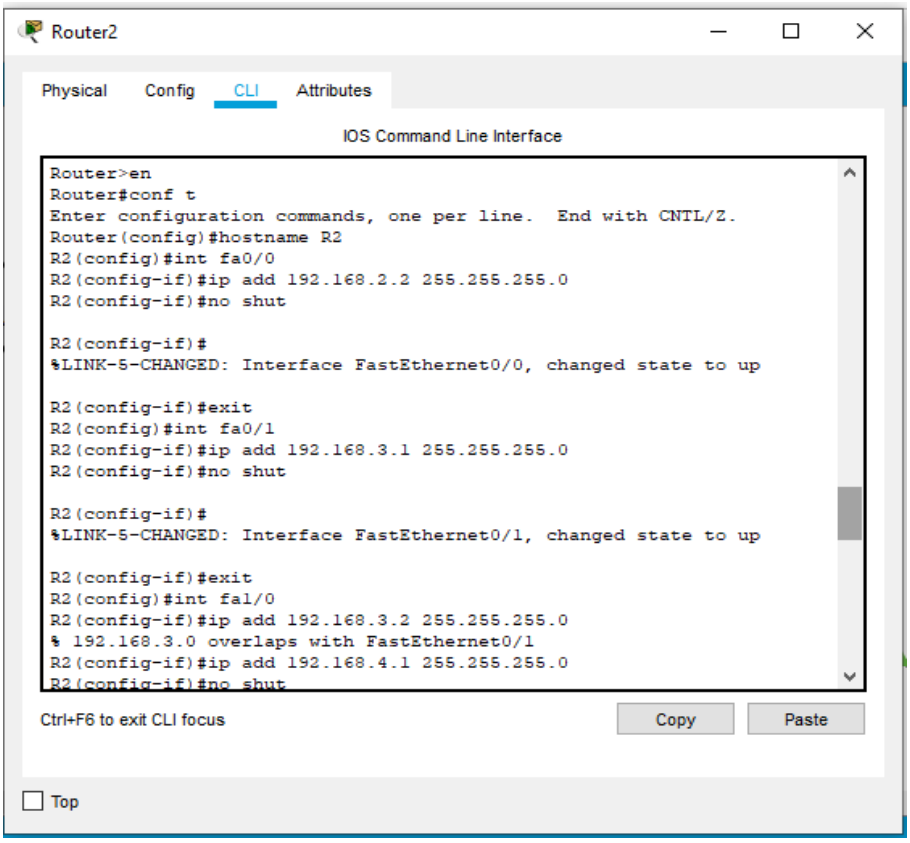


Figure IV- 3: Configuration des interfaces de router R2

On fait les mêmes étapes pour tous les routeurs restants.

Après cette étape, toutes les interfaces doivent passer à l'état ***up***. L'exécution de la commande **« sh ip int brief »** permet d'afficher l'adresse IP d'interfaces de router R2 et est-ce que le port est activé ou non !

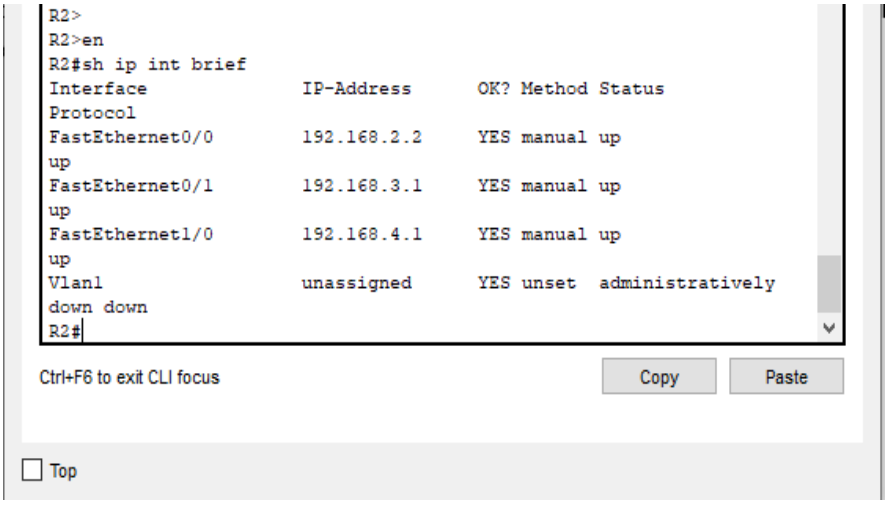


Figure IV- 4: Vérification des interfaces

IV.4.2.2 Configuration du protocole RIPv2

La configuration de ce protocole se fait en tapant les 3 commandes ci-dessous :

- Activation du protocole RIP dans les routeurs avec la commande suivante en mode configuration.

```
R1(config)#router rip
R1(config-router)#v 2
```

- Définition de tous les réseaux directement connectés à ce routeur avec la commande

```
R1(config-router)#network @réseau
```

On va configurer à présent les cinq routeurs.

Router R1 : les réseaux qui sont directement connectés au **routeur R1** sont 192.168.3.0/24, 192.168.4.0/24 et 192.168.2.0/24

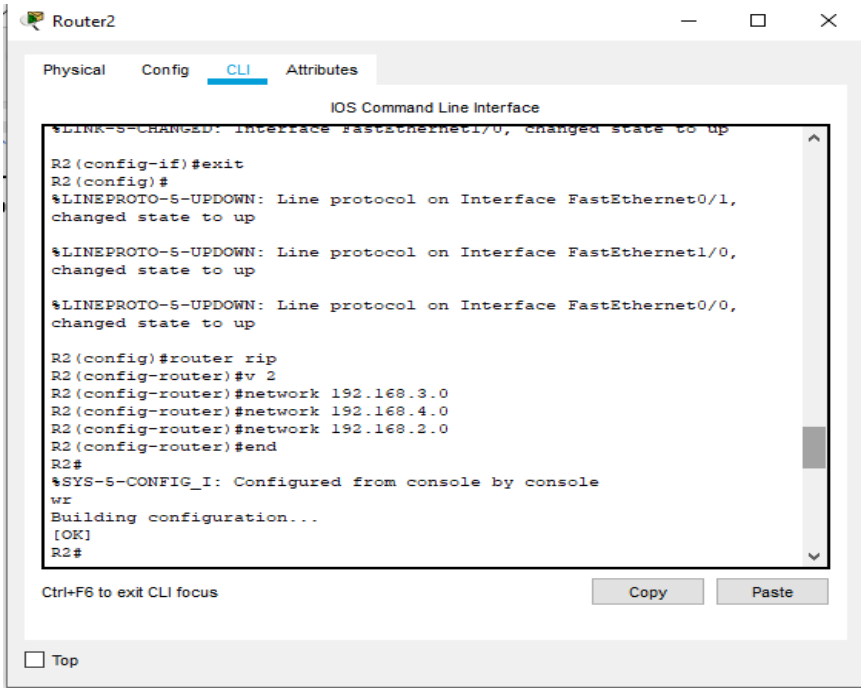


Figure IV- 5: Configuration du router rip

C'est tout concernant la configuration du protocole RIPV2. Nous ferons le test pour s'assurer que le routage est opérationnel.

- A présent, nous essayons de faire communiquer les routeurs entre eux : à partir du R4 vers R3, R4 vers R0, R3 vers R0.

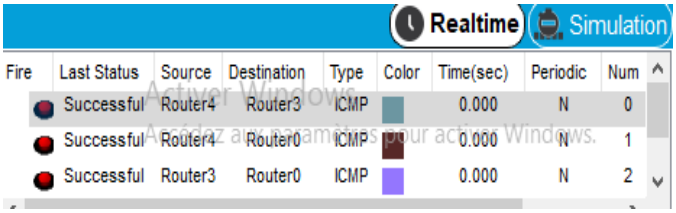


Figure IV- 6: Communication entre les routeurs

Ping de R4 vers l'interface de sortie des paquets R1 (192.168.5.1).

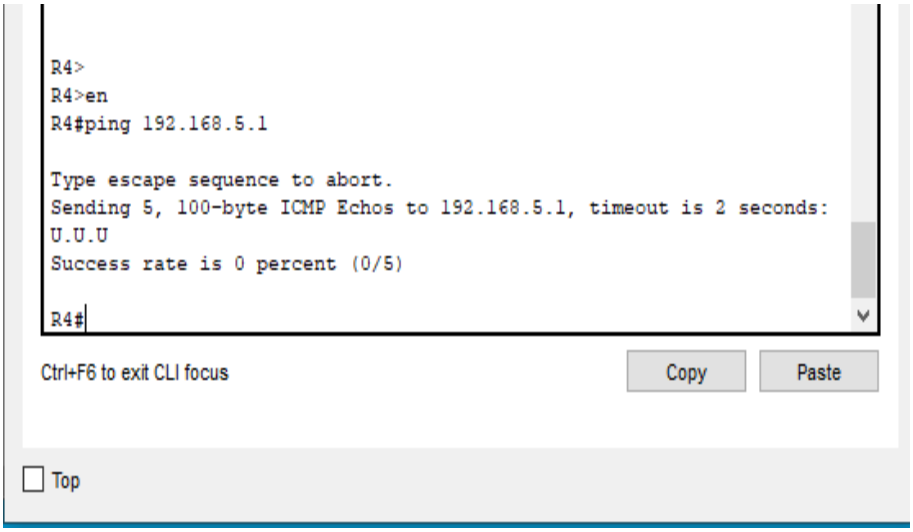


Figure IV- 7: Ping R4 vers R1

Problème 1: ping failed

Puisque le ping de R4 vers R0 est réalisé avec succès, donc il y a un problème sur le R1. La solution consiste à vérifier la configuration d'interfaces ou bien configuration de la définition du réseau c'est-à-dire network.

Solution01 : Vérification de la configuration de définition de réseau

On a oublié définir le réseau 192.168.5.0

```
R2(config)#router rip
R2(config-router)#v 2
R2(config-router)#network 192.168.5.0
R2(config-router)#exit
R2(config)#exit
```

On refait le ping.

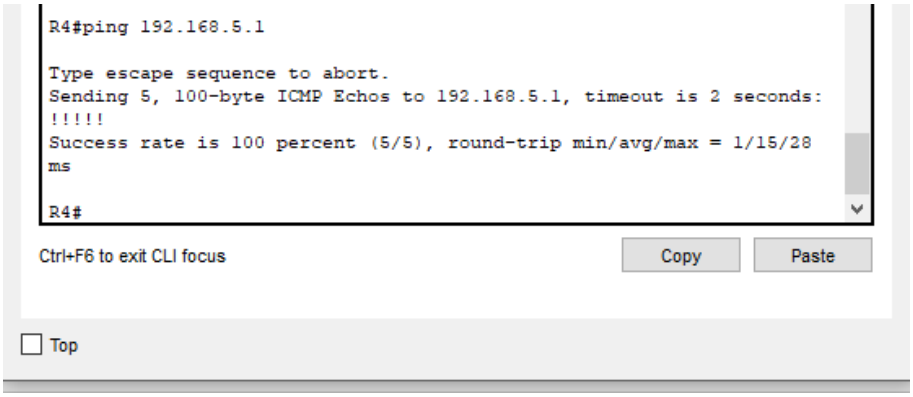


Figure IV- 8: Vérification de la communication entre R4 et R1

A l'instant présent, essayons de faire communiquer mutuellement les serveurs à partir du serveur Amazon. On lance un ping vers le serveur client.

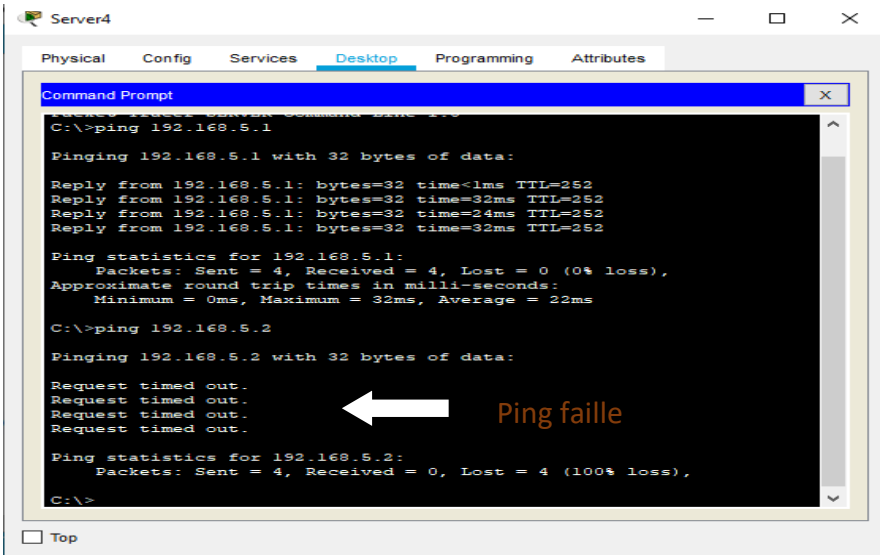


Figure IV- 9: Ping de serveur Amazon vers Serveur Client

Problème02 : ping échoué

Notre solution consiste à vérifier 2 trucs. La configuration sur le routeur R1 et la 2ème est la configuration du serveur client.

Solution01 : remplacez tous les serveurs avec des PC et on refait le test pour assurer la bonne configuration des routeurs.

Voici la nouvelle mise à jour de la topologie :

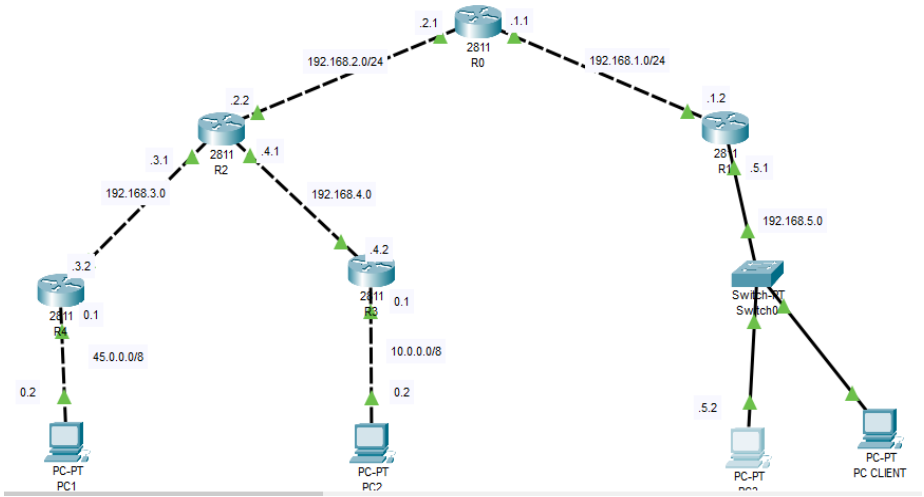


Figure IV- 10: Topologie d'essaie

On commence par configurer les PC.

PC1

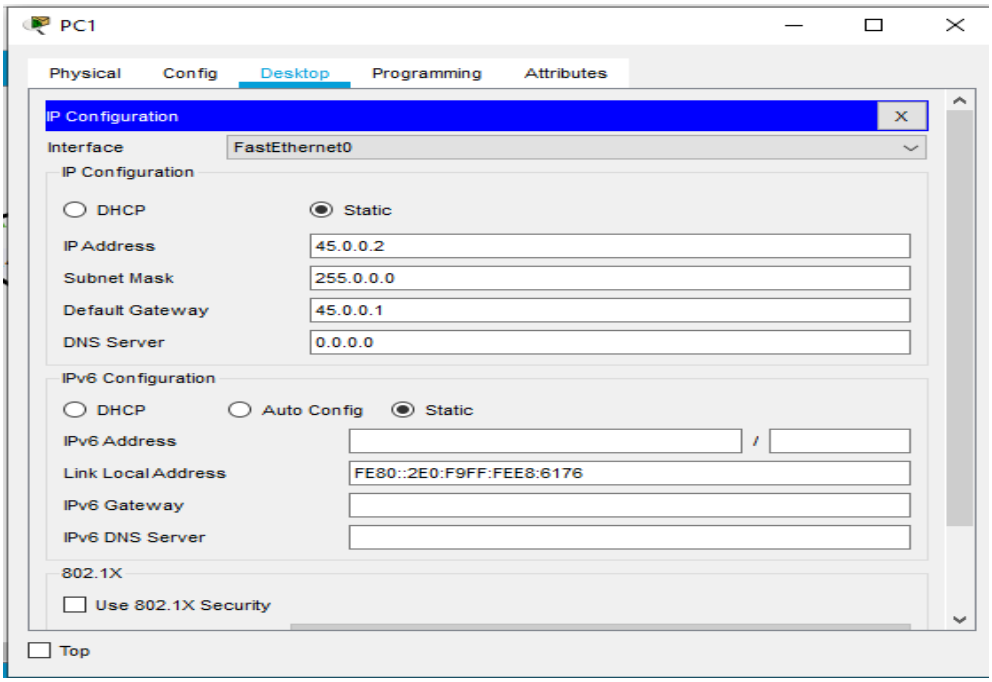


Figure IV- 11: Configuration du PC 1 (Amazon)

On fait la même chose pour les autre PC.

Essayons avec un autre Ping de PC1 vers PC 3, de PC1 vers PC2 et PC1 vers PC CLIENT

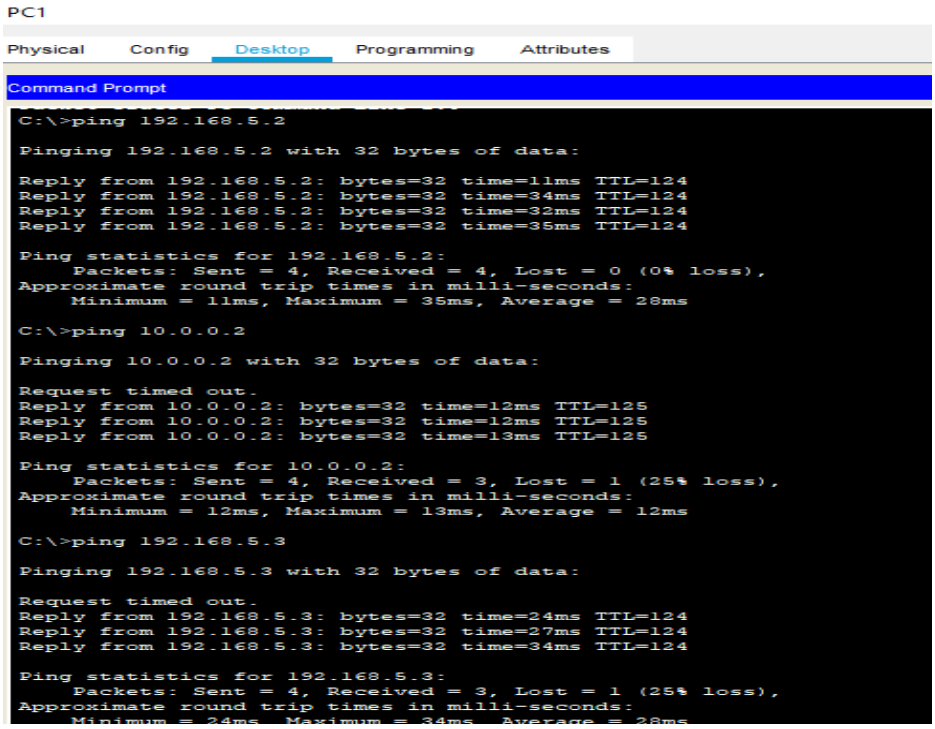


Figure IV- 12: Vérification de la communication entre les PCs

Après le ping, on a conclu qu'il y avait une mal configuration sur les serveurs et non pas sur le routage.

IV.4.2.3 Configuration des serveurs

Comme nous l'avons vu dans les étapes précédentes, l'adresse IP permet d'identifier une machine sur un réseau. Dans le cas d'un réseau IP (la majorité que vous rencontrerez et ceux qui nous intéressent), cette adresse est indispensable pour pouvoir communiquer avec les autres machines du réseau.

Nous allons nous intéresser ici à la manière dont cette adresse peut être obtenue. On distinguera deux méthodes, une manuelle, pour laquelle vous choisissez vous-mêmes l'adresse IP de votre machine et une dynamique où c'est un serveur qui vous fournit cette adresse. Ce serveur s'appelle un serveur DHCP et nous verrons qu'il a d'autres utilités que la simple distribution d'adresses IP.

Il existe donc **deux méthodes pour obtenir une adresse IP**. Cette dernière est configurée manuellement par l'utilisateur, ou dynamiquement par le serveur.

La méthode manuelle pose quelques problèmes de prime abord. En effet, vous avez vu que pour qu'une machine puisse communiquer avec ses voisines, son adresse IP devait se trouver dans le même réseau que les autres machines. Pour sortir du réseau local, il faut que notre machine connaisse l'adresse de la passerelle. Cela fait déjà quelques informations dont il faut avoir connaissance quand vous branchez votre ordinateur à un réseau local.

Serveur AMAZON :

- Affectation d'une adresse IP statique au serveur AMAZON

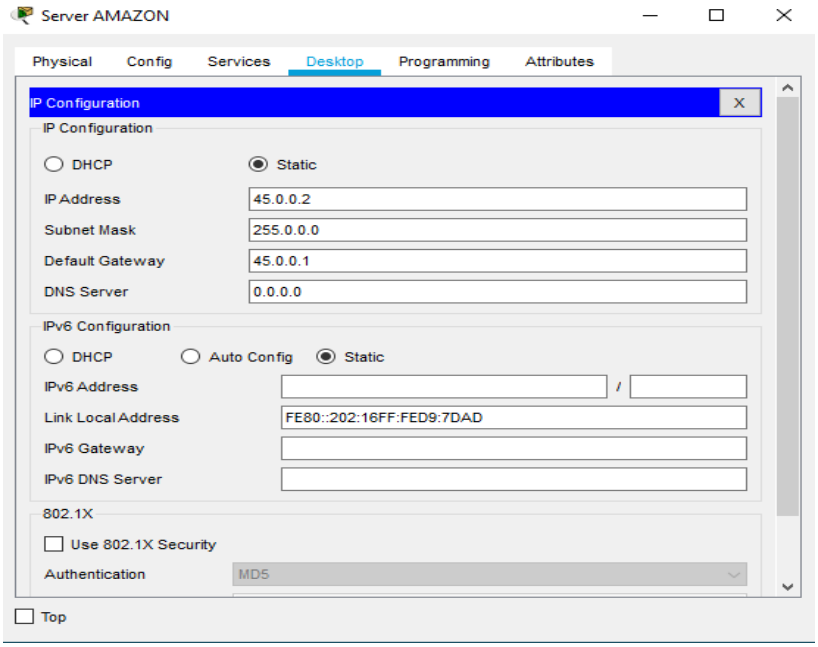


Figure IV- 13: Configuration de Serveur Amazon

- On va changer le contenu de la page web index.htp du serveur web et on doit vérifier ce changement à partir de la machine PC0.

Sélectionner le serveur WEB ==> Cliquer sur HTTP ==> sélection index.htp ==> cliquer sur edit

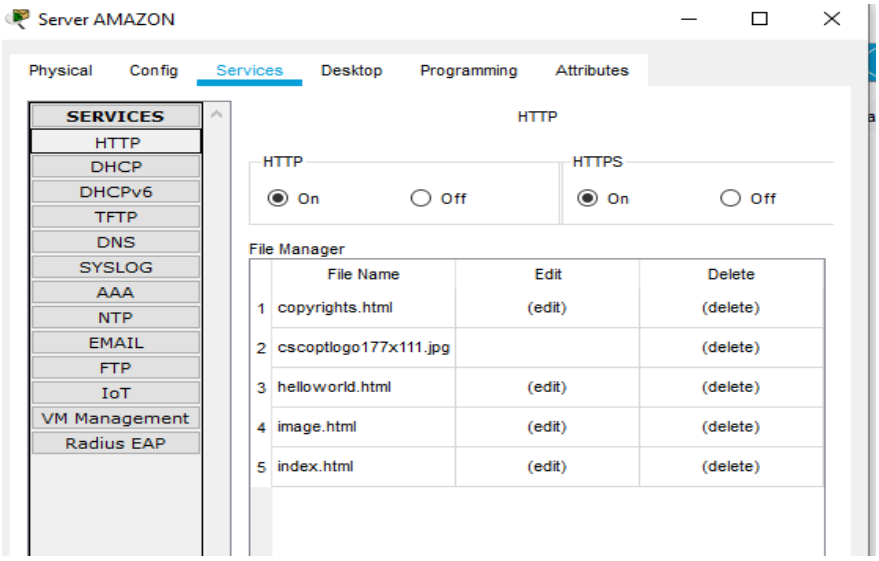


Figure IV- 14: Modification de la page Web d'Amazon

Ajouter un messager dans cette page dans code HTML

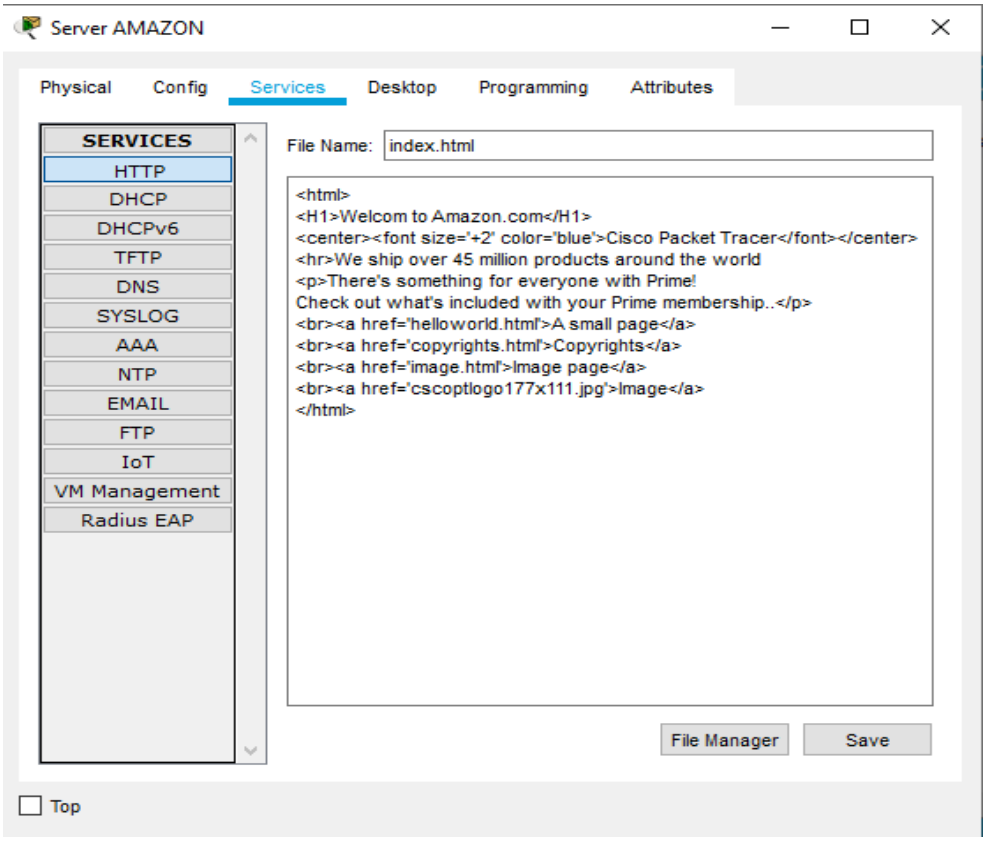


Figure IV- 15: Insertion de la Modification de la page Web d'Amazon

Vérification du changement à partir du Serveur CLIENT

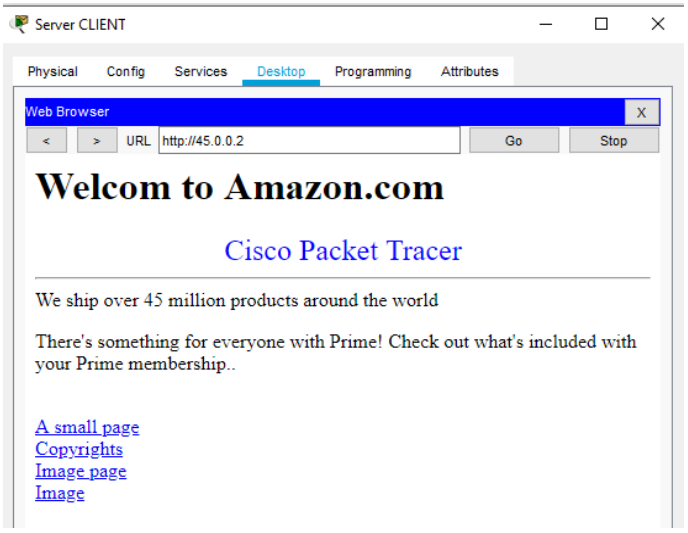


Figure IV- 16: Vérification de la Modification de la page Web d'Amazon

Serveur BANK :

- Affectation d'une adresse IP statique au serveur BANK

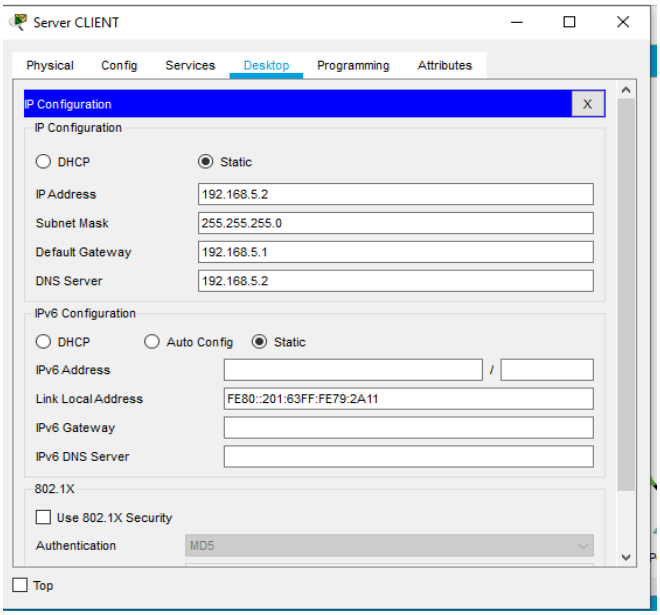


Figure IV- 17: Affectation des adresses IP au serveur BANK

Serveur Client :

- Autre problème, même si vous avez ces informations, comment vous assurez-vous que l'adresse IP que vous choisissez n'est pas déjà utilisée par une autre machine sur le réseau ?

On se rend donc bien compte qu'il serait bien d'avoir un mécanisme rapide et fiable pour adresser les machines d'un réseau. C'est là qu'entre en jeu le protocole DHCP.

Un protocole pour distribuer des adresses IP : La première fonction d'un serveur DHCP (*Dynamic Host Configuration Protocol*) est de fournir des adresses IP (associées à un masque, bien évidemment) aux machines en faisant la demande.

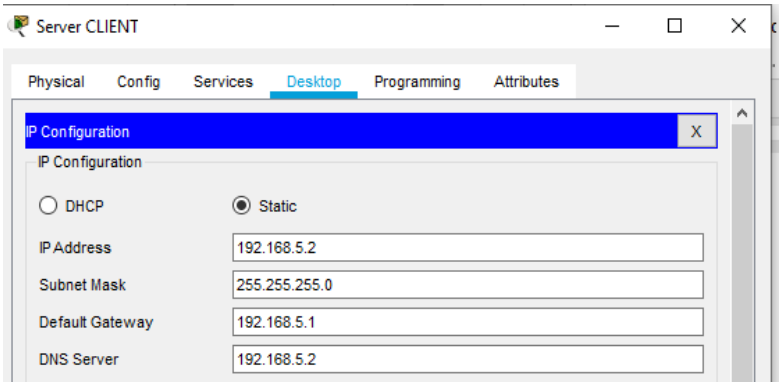


Figure IV- 18: Affectation des adresses IP au serveur Client

- Activation du protocole DHCP

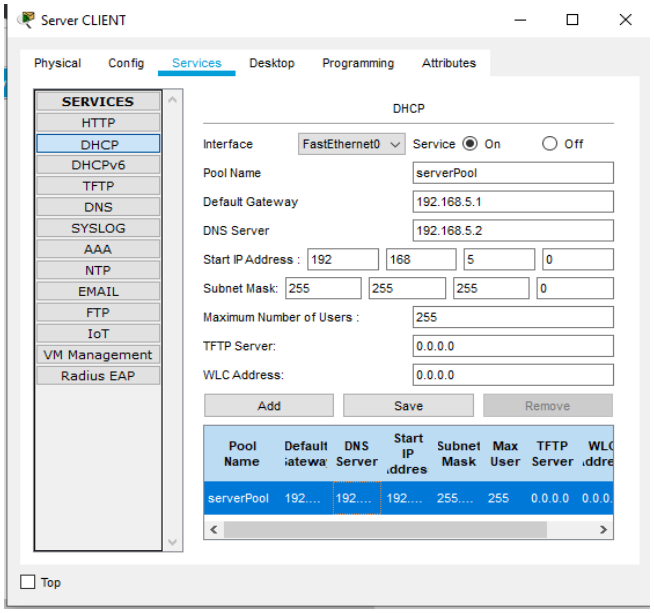


Figure IV- 19: Configuration du DHCP

On refait le test entre les serveurs s'ils fonctionnent ou non !

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	Server AMAZON	Server CLIENT	ICMP		0.000	N	0	(edit)	(delete)
	Successful	Server BANK	Server CLIENT	ICMP		0.000	N	1	(edit)	(delete)
	Successful	Server CLIENT	Server AMAZON	ICMP		0.000	N	2	(edit)	(delete)

Figure IV- 20: Test entre les serveurs

Donc, maintenant le problème est résolu

Un serveur DNS permet de faire l'association entre un nom de machine et une adresse IP, comme www.AMAZON.com est 45.0.0.2. Et www.Bank.com est 10.0.0.2

Configuration du DNS :

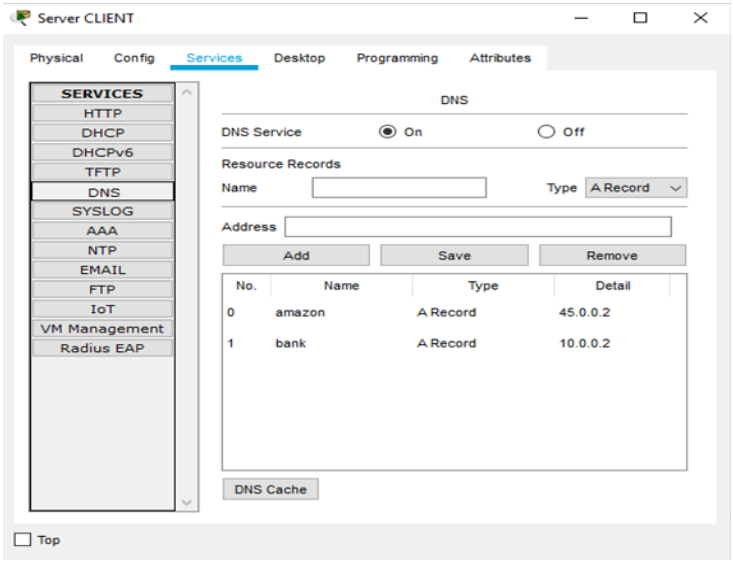


Figure IV- 21: Configuration d'un DNS serveur-client

Test du DNS :

Sélectionner le serveur WEB ==> Cliquer sur desktop ==> cliquer sur Web Browser ==> entrer le lien de la page souhaité

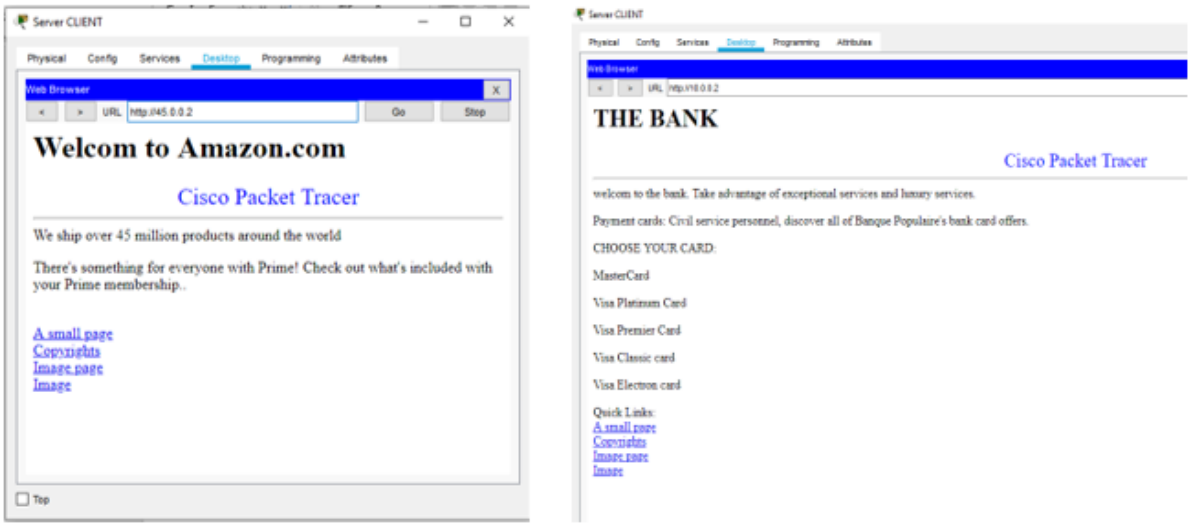


Figure IV- 22: Test d'un DNS serveur-client

IV.4.2.4 Configuration PC Client

Problème 3 : Comment allons-nous pouvoir joindre le serveur DHCP ?

- Tout ceci devrait nous rappeler quelque chose, non ? Le problème est le même que pour le protocole ARP.

Nous allons utiliser l'adresse de broadcast !

La trame permettant de trouver un serveur DHCP est une trame "DHCPDISCOVER", comme c'est un broadcast, elle est envoyée à l'adresse MAC ff:ff:ff:ff:ff:ff.

Comme la trame est envoyée en broadcast, le serveur DHCP doit obligatoirement se trouver dans le même réseau que la machine. Comme vous le savez, les routeurs (qui délimitent les réseaux) séparent les domaines de broadcast et ne relaient pas. Néanmoins, certains routeurs disposent de méthodes pour relayer ces trames DHCPDISCOVER. Mais, cela ne nous intéresse pas ici.

Une fois que notre serveur DHCP reçoit le DHCPDISCOVER, il va renvoyer une proposition, c'est un DHCPOFFER. Il va proposer une adresse IP, un masque ainsi qu'une passerelle par défaut.

Cette situation est illustrée par la figure ci-dessous :

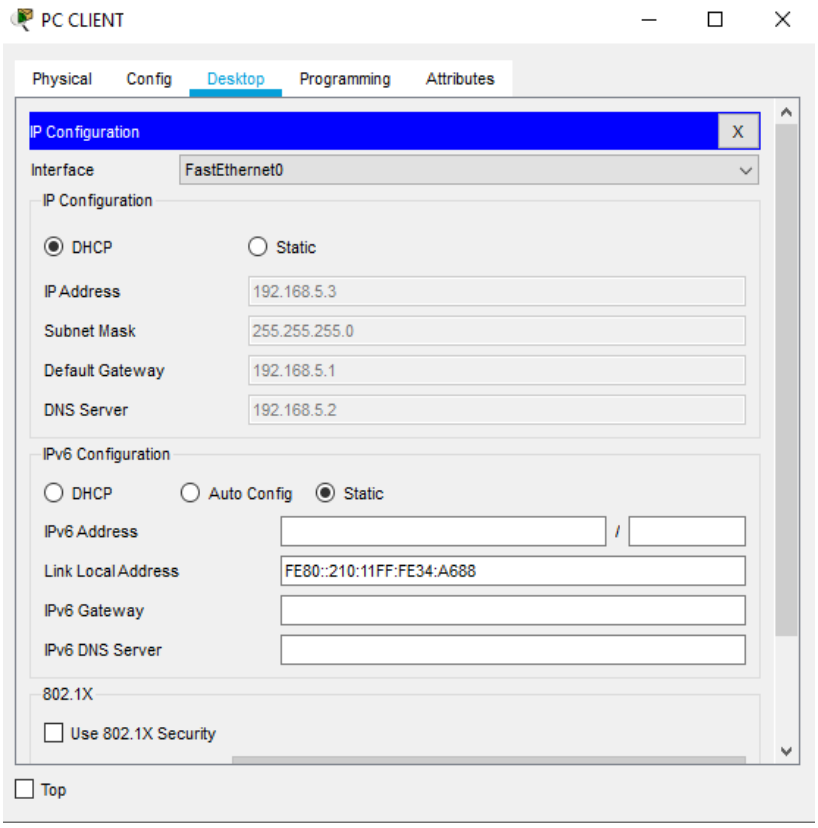


Figure IV- 23: Configuration du PC Client

En ce moment, on a terminé avec la création de la topologie de notre réseau, on va le tester !

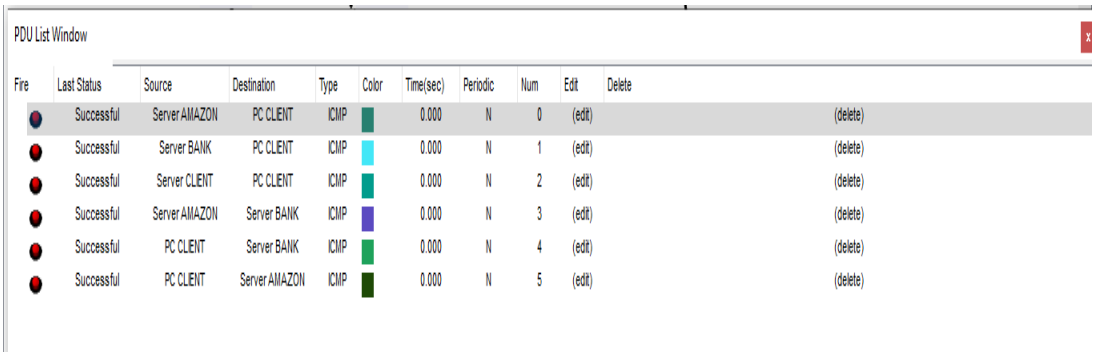


Figure IV- 24: Communication entre tous les machines

Tous marchent bien. On passe à la configuration du routeur R0 comme un **FAI** (Fournisseur d'Accès à Internet)

IV.4.3 Configuration d'un FAI DHCP sur un routeur Cisco

Dans cette topologie, on va configurer un DHCP sur le routeur R0 qui se situe sur internet beaucoup plus le comparer avec un FAI et on va configurer les routeurs R2 et R1 afin qu'ils reçoivent leurs adresses IP en provenance du FAI, donc du routeur R0.

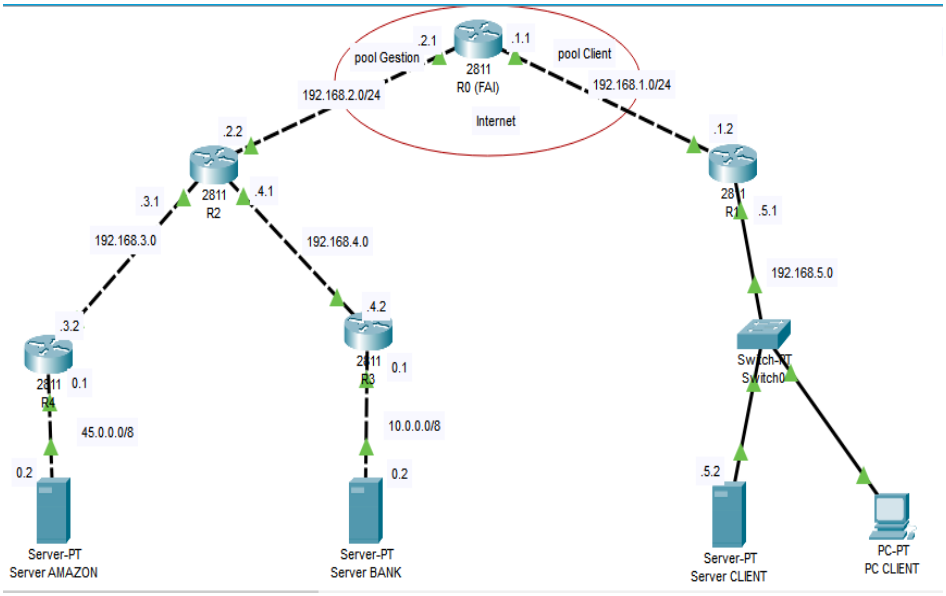


Figure IV- 25: Insertion d'un fournisseur d'accès à Internet

On va donc se connecter sur R0, la 1ère étape c'est donné un nom pour le pool du DHCP. L'un s'est pool Gestion et L'autre c'est la pool Client ; on lui assigne le réseau 192.168.2.0/24 pour pool Gestion avec une route par défaut qui sera l'interface vers le réseau privée 192.168.2.1 et le réseau 192.168.1.0/24 pour pool Client avec une route par défaut 192.168.1.1. En mode configuration globale, on va exclure les 100 premiers IP pour qu'on voit bien la modification car actuellement le routeur R2 a l'IP 192.168.2.2 et le routeur R1 a l'IP 192.168.1.2

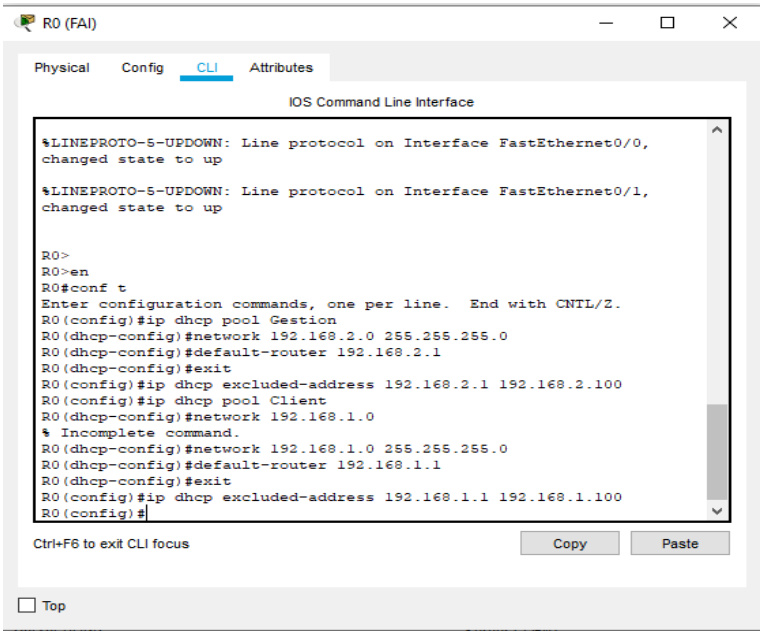


Figure IV- 26: Configuration d'un fournisseur d'accès à Internet

Ensuite, on va se connecter sur le routeur R2 ici sur l'interface FA0/0 avec la commande « **sh ip intbrief** ». On voit bien que le R2 avec l'interface Fa0/0 a une adresse 192.168.2.2 et le R1 a une adresse 192.168.1.1

En tapant « sh ip route », on voit aucune route configurée par défaut.

On va maintenant configurer les interfaces des deux routeurs en mode DHCP. Pour cela, il faut se connecter sur les interfaces « fa0/0 du R1 » et « fa0/0 du R2 ». Une fois connecté, il faut entrer la commande « **ip address DHCP** », puis on valide. En configurant les deux interfaces en mode DHCP, on peut voir un message qui va s'afficher sur la console indiquant que le DHCP est lui qui a fourni une adresse 192.168.1.101 sur le R1 et l'adresse 192.168.2.101 sur R2. En faisant un « **sh ip int brief** », nous confirmons l'affectation de la nouvelle IP.

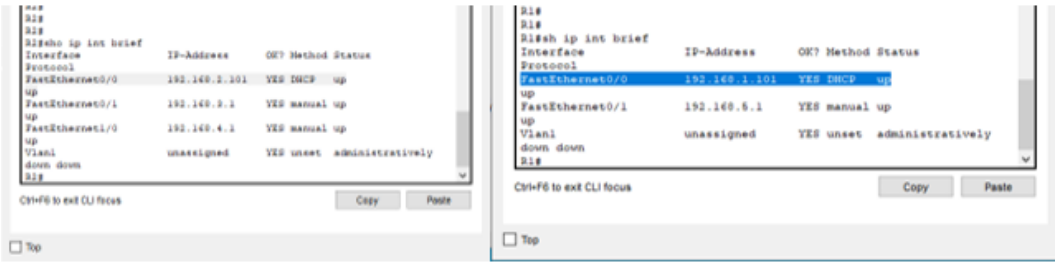


Figure IV- 27: Vérification des interfaces du FAI

Un « sh ip route » nous montre que le DHCP a bien envoyé sa paserelle par défaut.

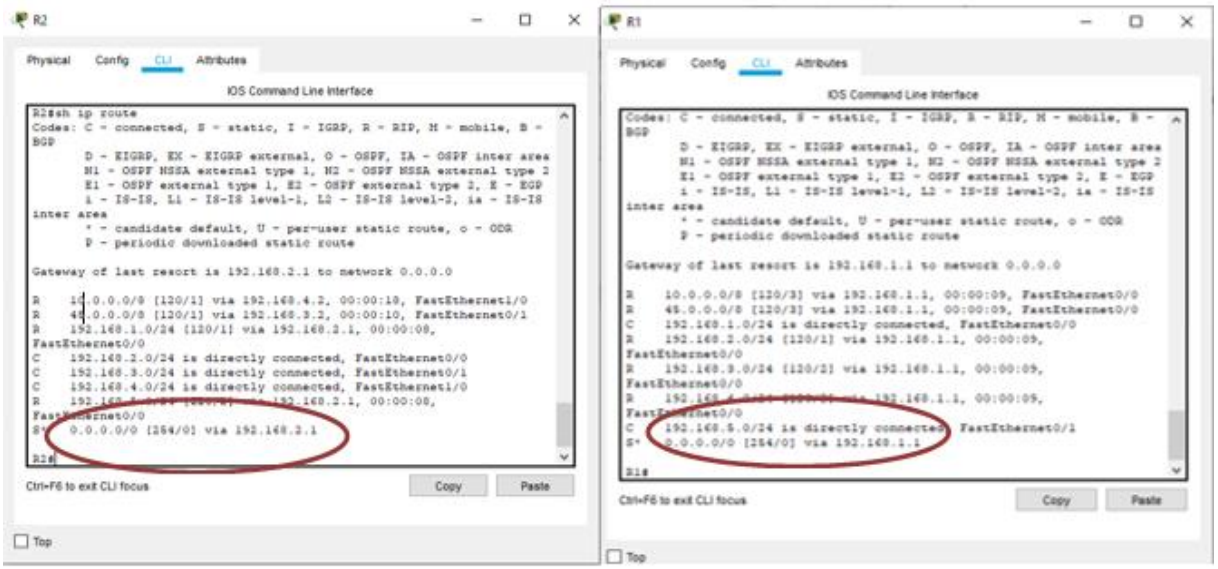


Figure IV- 28: Activation du DHCP sur R2 et R1

Si on essaye de faire un ping du serveur à l'autre bout, on constate que ça foctionne normalement et correctement ! on voit « percent 5/5 »

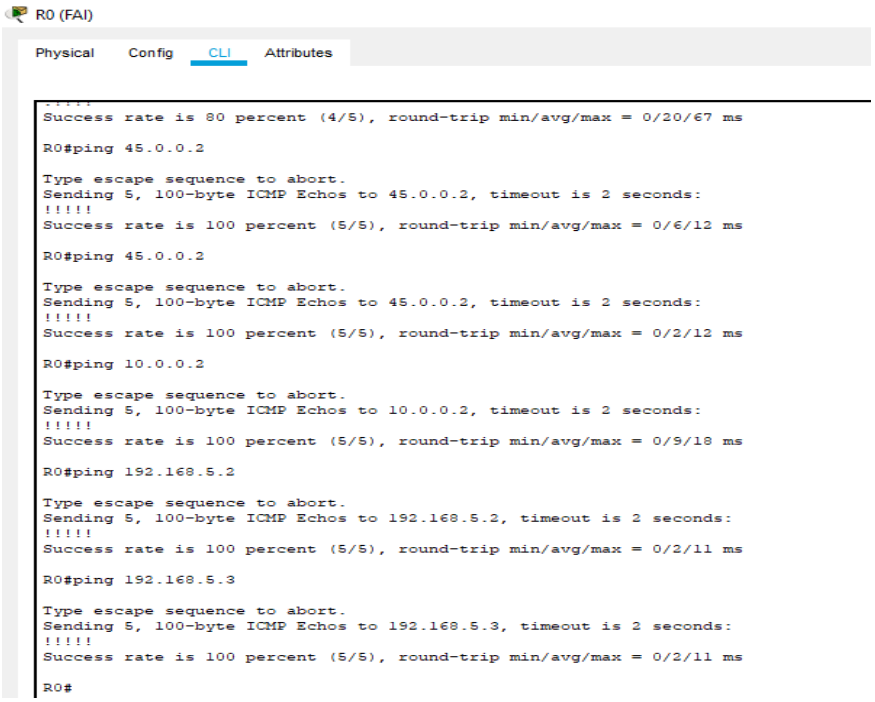


Figure IV- 29: Ping du R0 vers tous les hôtes

Le réseau fonctionne comme prévu. La source de trafic du R2 vers les destinations Internet est traduite tandis que le trafic provenant du R2 vers le R1 n'est pas traduit. Cependant, ce trafic doit être protégé lors de la traversée d'Internet public.

Problème 4 : La sécurité est un ensemble de stratégies, conçues et mises en place pour détecter, prévenir et lutter contre une attaque. Actuellement, il existe beaucoup de mécanisme de sécurité.

Solution : La solution d'interconnexion que fournit Internet pour répondre à ce besoin de communication sécurisé, consiste à utiliser les réseaux privés virtuels (VPN), qui sont idéals pour pouvoir exploiter au mieux les capacités du réseau Internet et de relier des sites à l'échelle de la planète en toutes sécurité. Au cours de ce chapitre nous présenterons les principales caractéristiques des VPN, à travers certaines définitions et principes de fonctionnement, les différentes typologies ainsi que les détails sur le protocole IPsec.

Un VPN (Réseau Privé Virtuel) nous permet de créer une connexion privée et sécurisée vers un autre réseau, généralement dans un autre pays. Cela donne l'apparence que nous sommes en train de naviguer depuis ce pays, en nous permettant d'accéder au contenu habituellement bloqué par la censure ou les autorités.

Votre connexion est protégée par un tunnel chiffré. Vos données ne peuvent donc pas être interceptées par des intrus, des pirates et des tiers. C'est la protection ultime dans la jungle numérique d'internet.

Alors que les VPN étaient initialement utilisés comme un moyen pour les employés d'accéder à distance et en toute sécurité à leur réseau de bureau.

IV.5 Implémentation du VPN

Cette partie montre comment configurer deux routeurs Cisco pour créer un tunnel VPN permanent de site à site sécurisé sur Internet, en utilisant le protocole IP Security (IPSec).

Les tunnels VPN IPSec peuvent également être configurés à l'aide de tunnels GRE (GenericRouting Encapsulation) avec IPsec. Les tunnels GRE simplifient considérablement la configuration et l'administration des tunnels VPN.

ISAKMP (Internet Security Association et Key Management Protocol) et IPSec sont essentiels à la construction et au chiffrement du tunnel VPN. ISAKMP, également appelé IKE (Internet Key Exchange), est le protocole de négociation qui permet à deux hôtes de s'accorder sur la manière de construire une association de sécurité IPsec. La négociation ISAKMP se compose de deux phases.

La **phase 1** crée le premier tunnel qui protège les messages de négociation ISAKMP ultérieurs.

La **phase 2** crée le tunnel qui protège les données.

IPsec entre alors en jeu pour crypter les données en utilisant des algorithmes de cryptage et fournit des services d'authentification, de cryptage et d'anti-répétition.

IV.5.1 Mise en place d'un VPN site à site

La nouvelle architecture proposée dispose de deux sites, notre topologie illustre leurs interconnexions via un tunnel VPN. Pour cela, il faudrait définir une clef partagée, une association de sécurité, une fonction de hachage . . . Ainsi, cette solution permettra aux sites RTARGA et RABOUDAOU d'échanger des données en passant par Internet d'une façon sécurisée en utilisant le tunnel VPN. Nous choisirons pour les deux sites les mêmes clefs de chiffrement, le type de hachage, la taille de police, la longueur des clés, la durée de vie de clé avant renégociation, la méthode de cryptage des données, la durée de vie de la clé de cryptage, une ACL permettant d'identifier le trafic à traiter par le tunnel et enfin la création d'une cryptomap.

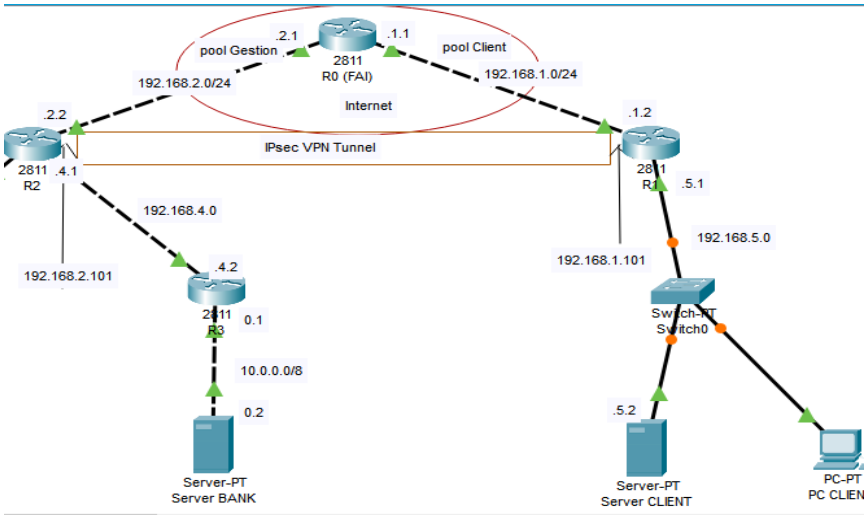


Figure IV- 30: Mise en place de tunnel

IV.5.2 Configuration du VPN

IV.5.2.1 Configuration ISAKMP

IKE n'existe que pour établir une SA pour IPSec. Il doit d'abord négocier cette SA (une SA ISAKMP) : les relations avec les routeurs des sites distants. Maintenant, nous allons commencer à travailler sur le site de la BANK(R2).

```
R2>ena
R2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#crypto isakmp enable
R2(config)#crypto isakmp policy 20
R2(config-isakmp)#encryption 3des
R2(config-isakmp)#hash md5
R2(config-isakmp)#group 1
R2(config-isakmp)#authentication pre-share
R2(config-isakmp)#lifetime 3600
```

Figure IV- 31: Configurer la politique de sécurité ISAKMP

IV.5.2.2 Protocole de gestion et l'échange des clés IPsec

La deuxième étape consiste à configurer la clef à l'aide de la commande suivante :

```
R2(config)#crypto isakmp key blockchain2020 add 192.168.1.101
```

Figure IV- 32: Architecture d'un VPN site à site

A chaque fois que la bank tentera d'établir un tunnel VPN avec le client, cette clé partagée (vpnkey) sera utilisée.

IV.3.2.3 Fonctionnement d'IPec

Pour configurer le protocole IPSec on a besoin de configurer les éléments suivants :

- Créer l'IPSec Transform.
- Créer une ACL étendue.
- Créer le crypto map.
- Appliquer crypto map à l'interface publique

Cette étape consiste à créer la transformation définie utilisée pour protéger les données

(IPSec) nommé ''vpntrans".

```
R2(config)#crypto ipsec transform-set myset esp-3des esp-md5-hmac
```

Figure IV- 33: Configurations d'IPsec

L'ACL étendu que l'on crée permettra de définir le trafic qui passera à travers le tunnel VPN. Dans notre projet, le trafic s'achemine du réseau 10.0.0.0/24 à 192.168.5.0/24.

```
R2(config)#access-list 100 permit ip 192.168.4.0 0.0.0.255 192.168.5.0 0.0.0.255
```

La crypto map est la dernière étape d'installation et d'établissement du lien entre ISAKMP définie précédemment et la configuration IPSEC :

```
R2(config)#crypto map mymap 20 ipsec-isakmp
% NOTE: This new crypto map will remain disabled until a peer
        and a valid access list have been configured.
R2(config-crypto-map)#
R2(config-crypto-map)#set peer 192.168.1.101
R2(config-crypto-map)#set transform-set myset
R2(config-crypto-map)#match add 100
R2(config-crypto-map)#exit
```

Figure IV- 34: Configuration de la crypto map

Maintenant il suffit d'appliquer la crypto map sur l'interface de sortie de routeur :

```
R2(config)#int fa0/0
R2(config-if)#crypto map mymap
*Jan  3 07:16:26.785: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is ON
R2(config-if)#end
R2#
%SYS-5-CONFIG_I: Configured from console by console
```

Figure IV- 35: Application de la crypto map

Dès que nous appliquons crypto map sur l'interface, nous recevons un message de Router qui confirme ISAKMP : ISAKMP is ON.

Les paramètres pour le routeur R1 sont identiques, la seule différence étant les adresses IP attribuées et les listes d'accès.

IV.5.2.4 Test de Protocol IPsec

À ce stade, nous avons terminé notre configuration et le tunnel VPN est prêt à être mis en place. Pour lancer le tunnel VPN, nous devons forcer un paquet à traverser le VPN et cela peut être réalisé en envoyant un ping d'un routeur à un autre.

Pour tester le VPN, on fait un ping entre les deux hôtes distants. Puis sur le routeur rattaché à la machine qui a effectué le ping, tapez la commande (show **crypto isakmp sa**) et on aura des informations sur la source et la distance.

```
R2#sh crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst             src             state          conn-id slot status
192.168.1.101   192.168.2.101   QM_IDLE           1062    0 ACTIVE

IPv6 Crypto ISAKMP SA
```

Figure IV- 36: Vérification des opérations d'ISAKMP

Et pour confirmer, on va voir les paquets encapsulés et autres en tapant (show **crypto ipsec sa).**

Figure IV- 37: Encapsulation des données

Cela prouve que le tunnel à bien été mis en place mais la communication entre les deux sites n'est pas sécurisée.

Problème 5 : pourquoi la communication n'est pas sécurisée malgré que le tunnel soit bien placé ?

Solution 01 : vérification si le crypto sur les interfaces du sortir de paquets est bien appliqué !

```
R2#sh crypto map
Crypto Map mymap 20 ipsec-isakmp
        Peer = 192.168.1.101
```

```
R1>en
R1#sh crypto map
Crypto Map mymap 20 ipsec-isakmp
        Peer = 192.168.2.101
```

Figure IV- 38: Vérification de la crypto map

- Les paramètres de cryptages sont bien appliqués
  - Notez que vous ne pouvez affecter qu'un seul crypto map à une interface.

Solution 02 : vérifier la liste d'accès ACL

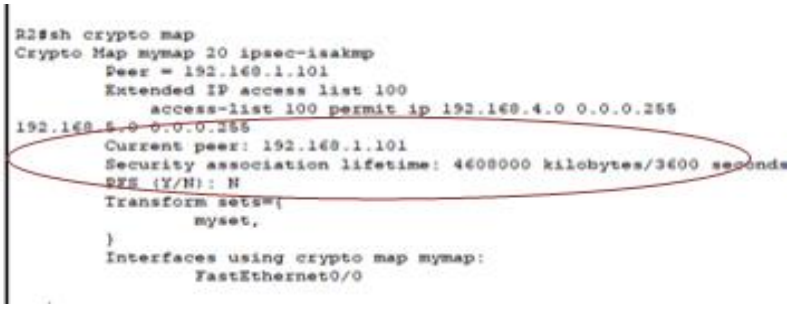


Figure IV- 39: Vérification de la liste d'accès

Il y a un problème sur la création de la liste d'accès pour sécuriser le réseau 192.168.4.0 du routeur et non pas le réseau où se trouve le serveur banque 10.0.0.0. Maintenant, on va juste créer une nouvelle liste d'accès permettant de crypter le réseau du serveur.

```
R2(config)#access-list 100 permit ip 10.0.0.0 0.255.255.255 192.168.5.0 0.0.0.255
```

Maintenant, il devrait être allumé on refait le test.

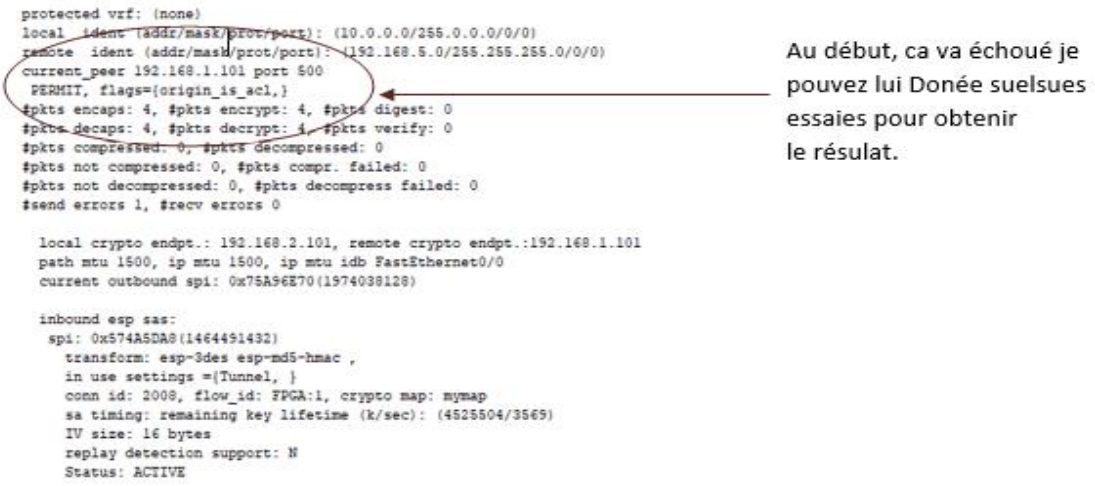


Figure IV- 40: Vérification d'IPsec sa

Maintenant, rappelez vous que le FAI n'a pas une idée sur les deux réseaux pourtant le ping marche bien. Avec le packet tracer, nous rentrons dans le mode de simulation. On laisse que le filtre ICMP, puis on va mettre en place un packet sur le serveur bank et puis un autre sur le serveur client et on capture et on observe le packet qui traverse les routeurs. Quand le packet arrive au FAI, on regarde à l'intérieur du packet.

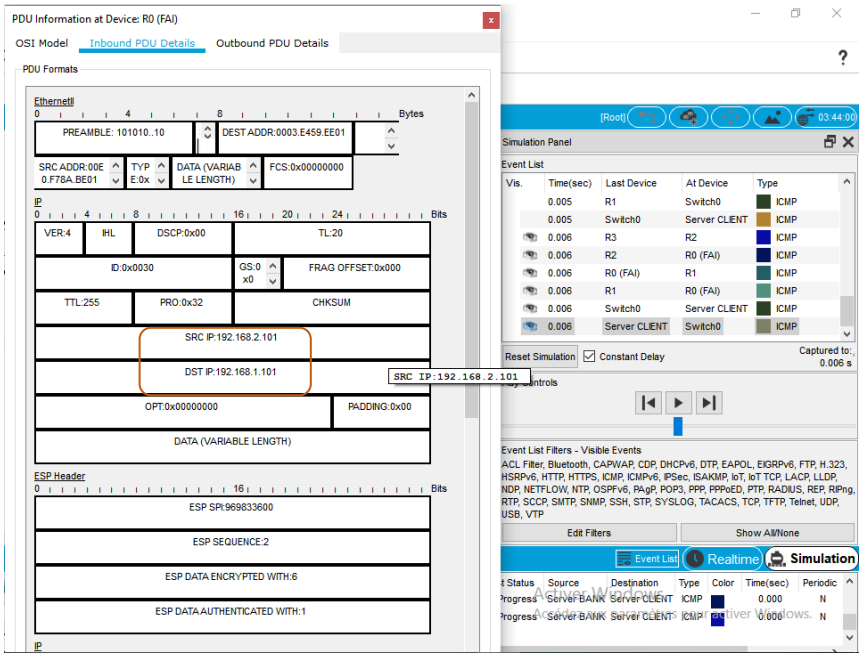


Figure IV- 41: Information du PDU sur R0(FAI)

Le FAI préocupe par le fait que le packet vient de la source IP 192.168.2.101 et avec une déstination 192.168.1.101 grace à l'en-tete ESP qui va crypter les données et les envoyer sur le tunnel .

Voici la réel source :

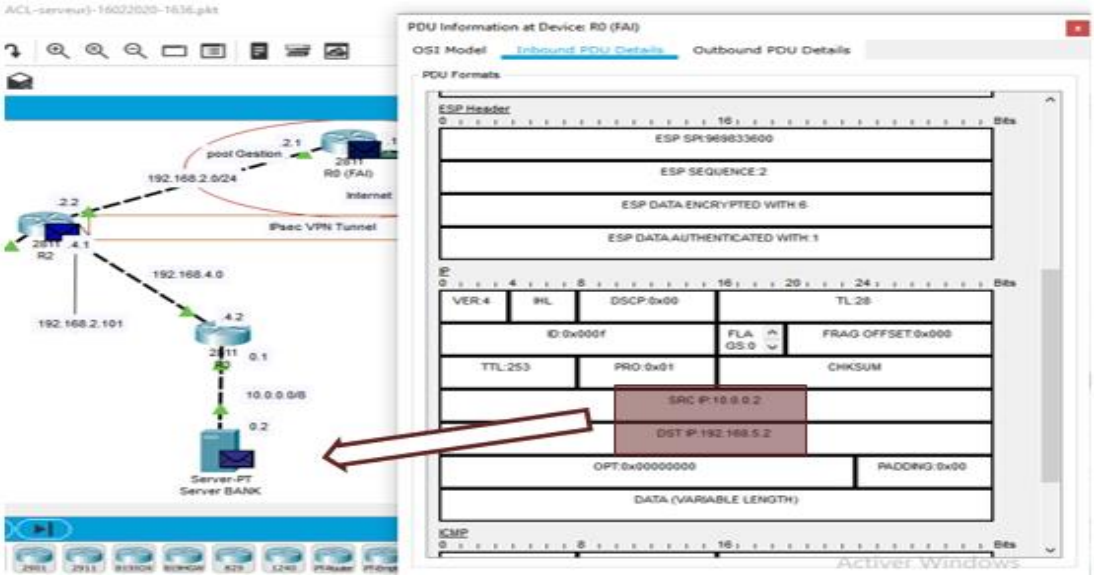


Figure IV- 42: Information du PDU non crypté

En fin, on a obtenu le meuilleur résultat .

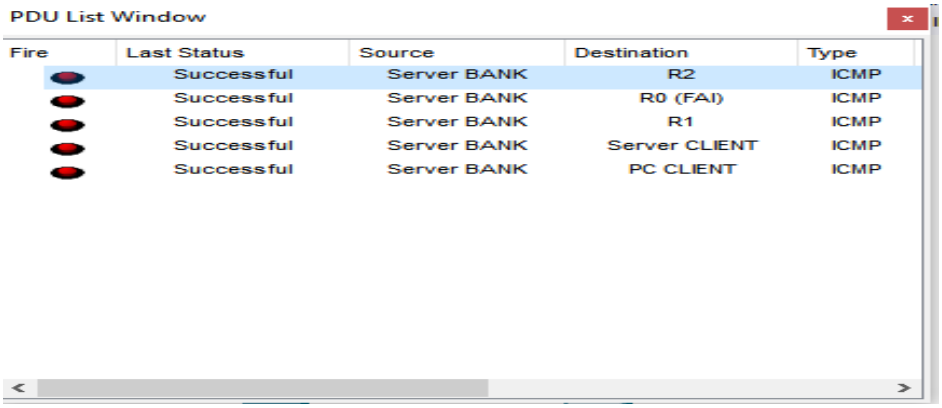


Figure IV- 43: Teste de la topologie

Avec un ping manuel maintenant

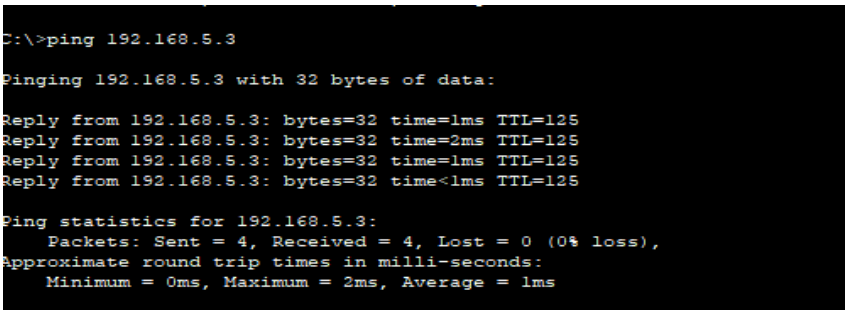


Figure IV- 44: Test de la topologie manuellement

Pour la confirmation, on tape la commande **sh crytpo ipsec sa**, qui nous retourne le résultat suivant :

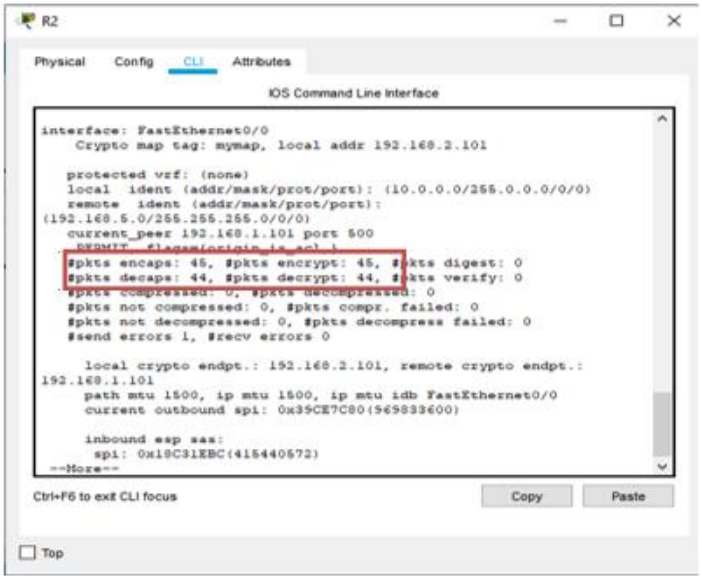


Figure IV- 45: Confirmation de cryptage et décryptage de données

Le réseau fonctionne comme prévu. La source de trafic du R2 vers les destinations Internet est traduite avec un mécanisme IPsec tandis que le trafic provenant du R4 vers le R3 n'est pas traduit. Cependant, ce trafic doit être protégé lors de l'échange de la monnaie. Donc, notre but est de protéger le trafic qui passe entre bank et amazon.

Problème 06 : comment protéger le trafic entre la bank et amazon ?

Solution 01 dans ce scénario, on propose de créer un tunnel GRE entre R4 et R3

IV.6 Implémentation d'in GRE tunnel

IV.6.2 Introduction

Tunneling fournit un mécanisme pour le transport de paquets d'un protocole à l'intérieur d'un autre protocole. Le protocole qui est transporté est appelé protocole passager, et le protocole qui est utilisé pour transporter le protocole passager est appelé protocole de transport. Generic Routing Encapsulation (GRE) est un des mécanismes de tunneling qu'utilise IP comme protocole de transport et il peut être utilisé pour transporter différents protocoles. Les tunnels se comportent comme des liens point à point virtuels qui ont deux extrémités identifiées comme tunnel source et tunnel destination.

Le diagramme suivant montre le processus d'encapsulation d'un paquet GRE quand il croise le routeur et rentre dans le tunnel d'interface :

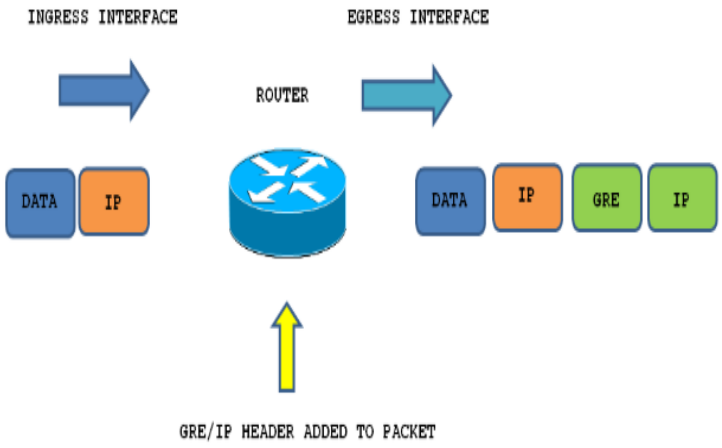


Figure IV- 46: Fonctionnement du Tunnel GRE

Pour notre configuration, on a utilisé bien la topologie en capture d'écran suivante :

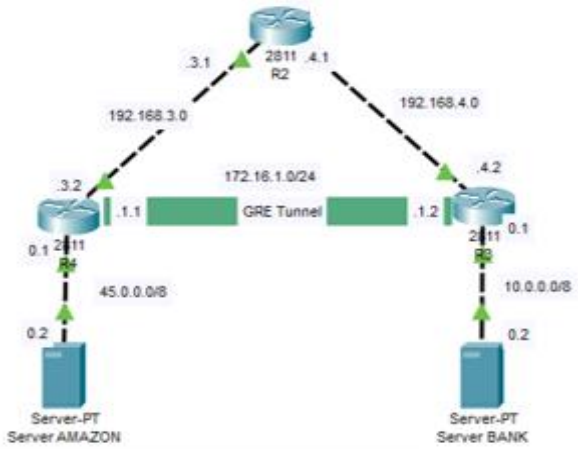


Figure IV- 47: Insertion du Tunnel GRE

IV.6.2 Configurons un tunnel GRE :

Configurer un tunnel GRE revient à configurer un tunnel d'interface, qui est une interface logique. Après, il faut configurer les extrémités pour l'interface de tunnel.

Pour configurer la source et la destination du tunnel, vous devez configurer les commandes de tunnel source {ip-address | type d'interface} et du tunnel destination {nom-hôte | ip-address} dans le mode de configuration de l'interface du tunnel.

La configuration suivante explique comment créer un tunnel GRE simple entre deux points et les étapes nécessaires pour vérifier la connexion entre deux réseaux. Les sous-réseaux de R4 et R3 (respectivement 45.0.0.0/8 et 10.0.0.0/8) communiquent entre eux via un tunnel GRE sur Internet. Les deux interfaces du tunnel font parties du réseau 172.16.1.0/24.

La première étape consiste à créer l'interface de tunnel sur les deux routeurs.

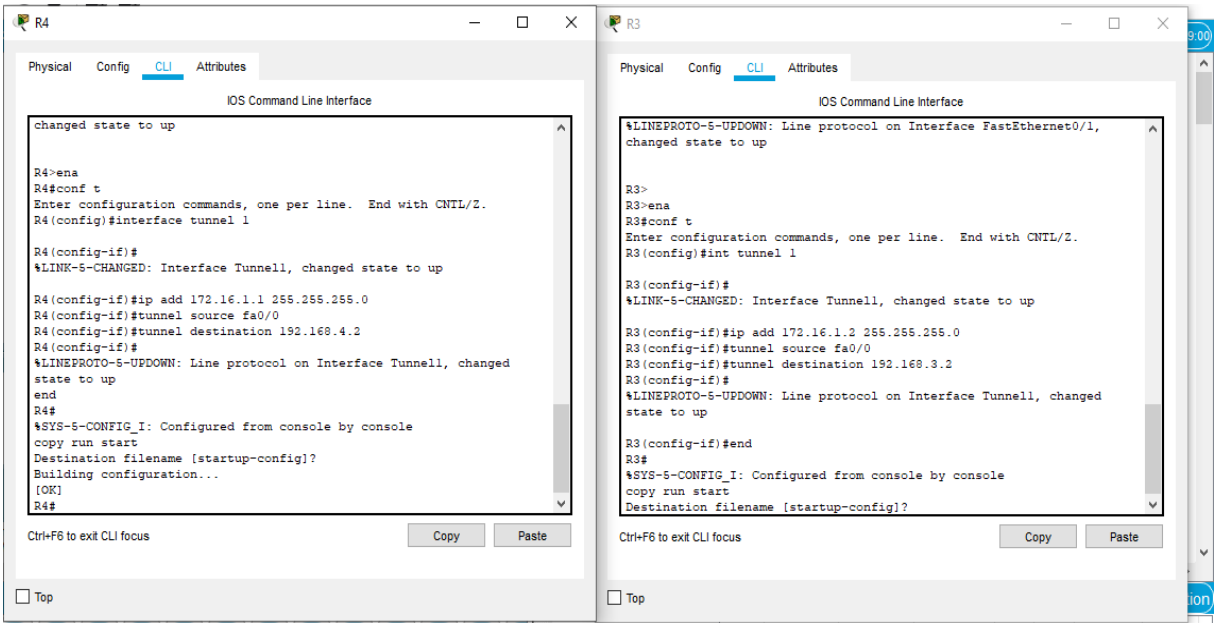


Figure IV- 48: Création d'interfaces de Tunnel

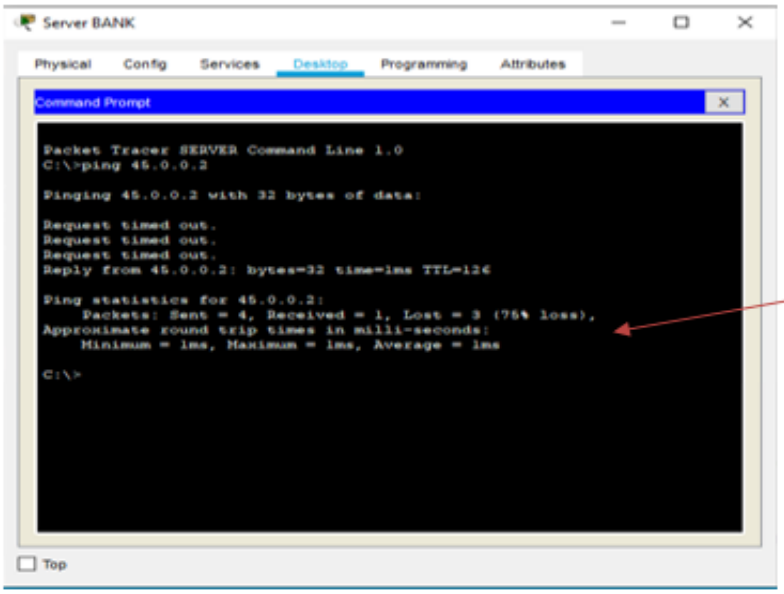
Les hôtes des deux réseaux privés ne pourront pas se joindre que si les protocoles de routage ou les routes statiques sont configurés dans les routeurs.

```
R4(config)#ip route 10.0.0.0 255.0.0.0 172.16.1.2
R3(config)#ip route 45.0.0.0 255.0.0.0 172.16.1.1
```

Maintenant, tous les deux réseaux privés (10.0.0.0/8 et 45.0.0.0/8) peuvent communiquer entre eux à travers du tunnel GRE.

IV.6.3 Test de tunnel

Vérifions ce tunnel GRE avec un ping du serveur BANK vers le serveur AMAZON



Comme vous pouvez le voir c'est réussi

Figure IV- 49: Vérification si le Tunnel est bien créé

On peut aussi utiliser la commande "tracert" pour qu'on peut voir le chemin du trafic

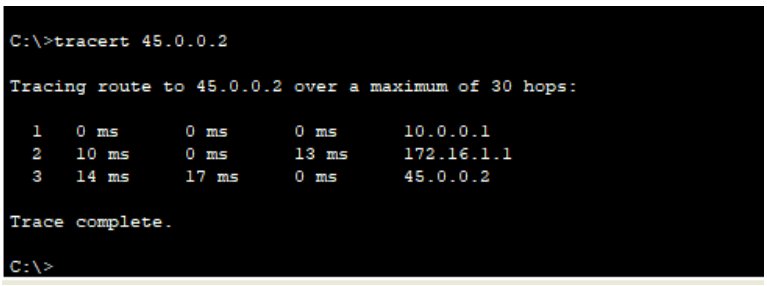


Figure IV- 50: Chemin du traffic

Notre trafic va aller en premier par l'adresse du l'interface 10.0.0.1 puis par l'interface logique 172.16.1.1 du tunnel GRE après par l'adresse de destination 45.0.0.2

Partie 2 : Construire une blockchain

IV.7 Introduction

Dans le chapitre 2**,** nous avons appris ce qu'est une blockchain et comment elle fonctionne. Dans de plus, nous avons appris dans le chapitre 3 l'architecture d'un réseau. Dans ce chapitre, on va commencer à construire la blockchain et tous ses fonctionnalités. Commençons par crée notre projet ou nous allons taper notre code pour crée la blockchain, puis création de la structure de données blockchain utilisant un constructeur puis nous ajouterons beaucoup de différents types de fonction de notre blockchain par l'ajoute de différentes méthodes à son prototype. Telles que la création des blocs et les transactions,

ainsi que la capacité de hachage des données et des blocs. Nous lui donnerons également la possibilité de faire une preuve de travail et de nombreuses autres fonctionnalités qu'une blockchain devrait être capable de faire. Nous nous assurons ensuite que la blockchain et entièrement fonctionnel en testant les fonctionnalités ajoutées au fur à mesure de notre progression. Donc en construisant blockchain étape par étape, pour nous gagne une meilleure compréhension de la façon de la blockchain fonctionne.

IV.8 Configuration du projet

Commençons par construire notre blockchain projet. La première chose que nous allons faire est ouvrir notre terminal et créer notre blockchain en tapant des commandes dans le répertoire terminal.

Etape 01 : commençons par crée un dossier appelé blokchchain2020, dans ce dossier créons un répertoire appelé programmation_BC. Ce répertoire est actuellement vide. À l'intérieur de ce répertoire programmation_BC se trouve ou nous allons faire tout notre programmation. Nous allons construire toute notre blockchain à l'intérieur de ce répertoire. Comme illustre la figure suivante :

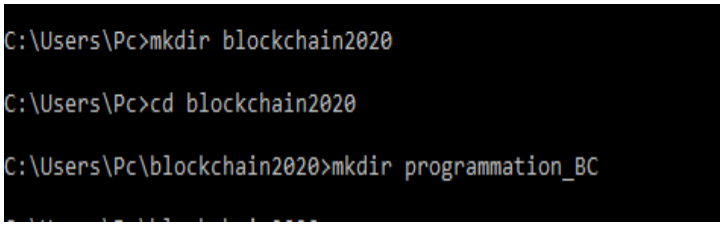
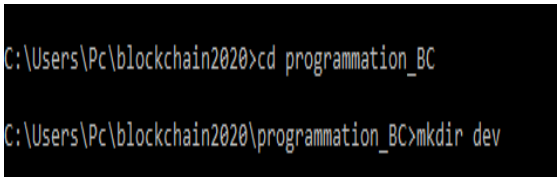


Figure IV- 51: Créations de notre projet

Maintenant, notre répertoire programmation_BC est prêt, et la première chose que nous devons faire est d'ajouter un dossier et fichiers en elle. Le premier dossier nous voulons mettre sera appelée dev. En tapant les commandes suivantes :



Dans ce répertoire, nous allons construire nos données blockchain, nous allons créer deux fichiers blockchain.js et test.js. Pour faire cela, entrez les commandes suivantes :

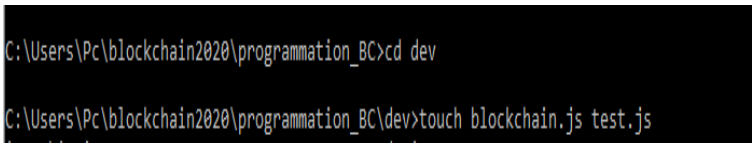


Figure IV- 52: Créations des fichiers avec les extensions "js"

La commande « touch » permet d'ouvrir un fichier s'il existe ou de le crée s'il n'existe pas. On peut n'utilise pas cette commande « touch » voici une autre méthode. Dans ce dernier dossier on ouvre l'éditeur de texte en tapant les codes et les enregistres avec les extensions "js" (test.js et blockchain.js).

Le fichier blockchain.js est l'endroit où nous allons taper notre code pour crée la blockchain et le test.js est l'endroit où nous allons écrire notre code pour tester la blockchain.

Ensuite, revenons à notre répertoire programmation_BC en tapant la commande suivante dans le terminal : cd..

Dans le répertoire programmation_BC, exécutons la commande suivante pour crée le npm projet :

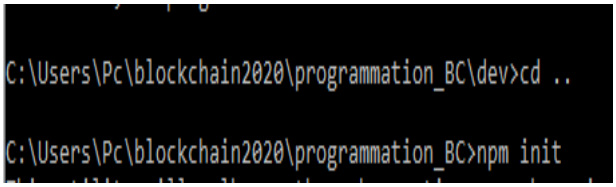


Figure IV- 53: Création d'un fichier package.json.

Après avoir exécuté la commande précédente, on obtiendra quelques options sur notre terminal.

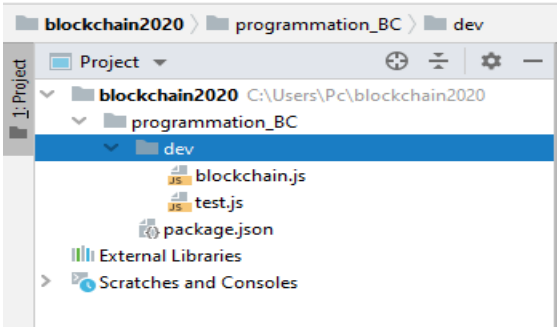


Figure IV- 54: Création de projet est prête

Ce fichier. json gardera une trace de notre projet toutes les dépendances dont nous avons besoin, permet nous pour exécuter des Scripts. Nous nous travaillons plus à l'intérieur de ce fichier package.json

IV.9 Programmation de la blockchain

IV.9.1 Introduction

Dans cette section ; nous couvrirons les points suivants :

- Apprendre a créé une fonction de constructeur de la blockchain
- Construire et tester les différentes méthodes
- Comprendre la mise en œuvre de la PoW
- Crée et tester un genesis block

Alors, commençons !

IV.9.2 Création de la blockchain

Exigences : J'ai construit cela dans Node.js, mais vous pouvez facilement le construire sur Repl.it si vous préférez. Nous utiliserons également le package sha256 npm pour hacher les blocs.

Si vous êtes nouveau sur Node.js, tout d'abord, vous devrez créer un nouveau répertoire, créer un index.js fichier, puis à partir de votre terminal, exécuter à npm init -y partir de votre nouveau répertoire. Cela crée un fichier package.json, qui nous permettra d'installer le package sha256. Exécutez npm i sha256 pour installer la bibliothèque sha256. Si vous êtes sur Repl.it, créez un nouveau replNodejs, installez le js-sha256package et exigez-le.

Commençons par construire notre blockchain Structure de données. Nous allons commencer par ouvrir tous les fichiers que nous avons dans notre blockchain en utilisant l'éditeur Sublime. Si vous êtes à l'aise avec n'importe quel autre éditeur, vous peut aussi l'utiliser. Ouvrez toute notre blockchain répertoire dans l'éditeur que vous préférez.

Nous allons construire notre structure de données blockchain dans le fichier dev /blockchain.js que nous avons créé dons la section précédente, Configuration du projet. Construisons cette structure de données blockchain en utilisant une fonction constructrice. Alors, commençons :

- Maintenant que nous sommes configurés, il est temps de créer une nouvelle classe, nous avons donc un modèle que nous utiliserons pour construire chaque bloc. Chaque bloc de notre blockchain simple contiendra 3 éléments de données : l'index du bloc, la valeur de hachage du bloc qui le précède et un hachage de lui-même.
- Pour créer notre bloc, j'utilise une classe JavaScript, mais une fonction constructeur fonctionnerait tout aussi bien car c'est vraiment tout ce qu'une classe est sous le capot. J'ai défini la fonction constructrice pour prendre toutes nos données à l'exception du hachage comme paramètres. Le bloc examinera alors l'index, l'horodatage, les données et le hachage précédent, et générera son propre hachage ou empreinte digitale.
- Continuons à construire notre structure de données blockchain. Après avoir défini notre fonction constructeur dans la section précédente, la suivante chose que nous voulons faire avec notre fonction constructeur est de placer une méthode dans notre fonction Blockchain. Cette méthode que nous allons créer s'appellera createNewBlock. Comme son nom l'indique, cette méthode va créer un nouveau bloc pour nous. Suivons les étapes mentionnées ci-dessous pour construire notre blockchain :

IV.9.2.1 La création d'ajoute un nouveau bloc

```
blockchain.js   test.js   package.json

function Blockchain () {
    this.chain = [];
    this.newTransactions = [];

}

Blockchain.prototype.createNewBlock = function (nonce, previousBlockHash, hash) {
    const newBlock = {
     index: this.chain.length + 1,
     timestamp: Date.now(),
     transactions:
     this.newTransactions,
     nonce: nonce,
     hash: hash,
     previousBlockHash: previousBlockHash,
    };
    this.newTransaction = [];
    this.chain.push(newBlock);
    return newBlock;

}
```

Figure IV- 55: Programmations d'ajoute d'un nouveau bloc

En ajoutant ces deux dernières lignes de code, notre méthode createNewBlock est prête. Fondamentalement, c'est qu'elle crée un nouveau bloc. À l'intérieur de ce bloc, nous avons nos transactions et les nouvelles transactions qui ont été créées depuis l'extraction de notre dernier bloc. Après avoir créé un nouveau bloc.

- *Test de la méthode constructeur de la blockchain :*

La première chose que nous devons faire est d'exporter notre fonction constructeur Blockchain car nous allons utiliser cette fonction dans notre fichier test.js. Donc, pour exporter la fonction constructrice, nous allons aller au bas du fichier blockchain.js, taper la ligne de code suivante, puis enregistrer le fichier.

```
module.exports = Blockchain;
```

Figure IV- 56: Exportations de la fonction constructrice

Ensuite, accédez au fichier dev / test.js, car c'est ici que nous testerons notre méthode createNewBlock. Maintenant, la première chose que nous voulons faire dans notre fichier dev / test.js est d'importer notre fonction constructeur Blockchain, alors tapez ce qui suit :

```
blockchain.js   test.js

const Blockchain = require('./blockchain');
const bitcoin = new Blockchain();


console.log(bitcoin);

```

Figure IV- 57: Configuration de projet bitcoin

Allez maintenant dans notre fenêtre de terminal. Ici, nous sommes actuellement dans le répertoire blockchain, et notre fichier test.js est dans notre dossier dev, alors tapez la commande suivante dans le terminal:node test.js

Nous observerons la Blockchain dans la fenêtre du terminal, comme le montre la capture d'écran suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
Blockchain { chain: [], newTransactions: [] }
```

Figure IV- 58: Création de projet bitcoin

Nous pouvons observer que Blockchain a une chaîne vide et un tableau de transactions vide. C'est exactement ce que nous attendions de la sortie.

- *Test d'ajoute des nouveaux blocs :*

Nous ajouterons la ligne de code suivante : **bitcoin.createNewBlock();**

Ensuite, nous allons créer un hachage pour notre précédent BlockHash, suivi par un autre hachage pour notre paramètre de hachage, comme suit :

bitcoin.createNewBlock(1998,'KAWTER','BO14UD05GH19en98');

Donc, maintenant on a créé notre bitcoin. Enregistrez ce fichier et exécutez notre fichier test.js à nouveau dans le terminal. Nous aurons ensuite à observer la sortie suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
Blockchain {
  chain: [
    {
      index: 1,
      timestamp: 1586284300109,
      transactions: [],
      nonce: 1998,
      hash: 'BO14UD05GH19en98',
      previousBlockHash: 'KAWTER'
    }
  ],
  newTransactions: [],
  newTransaction: []
}
```

Figure IV- 59: Test d'ajoute un nouveau bloc

Dans la capture d'écran précédente, nous avons observé toute la structure de données de la chaîne de blocs dans la chaîne. Ce bloc a également le hachage, nonce et previous Block Hash paramètres que nous avions saisis. Il n'a aucune transaction parce que nous n'en avons pas créé transactions encore. Par conséquent, nous pouvons conclure que la méthode createNewBlockfonctionne très bien.

Maintenant, testons encore plus notre méthode en créant quelques blocs de plus dans notre chaîne. Comme elle montre la figure suivante :

```
blockchain.js   test.js   package.json

1  const Blockchain = require('./blockchain');
2  const bitcoin = new Blockchain();
3  //bitcoin.createNewBlock();
4  bitcoin.createNewBlock(1998,'KAWTER','BO14UD05GH19en98');
5  bitcoin.createNewBlock(1997,'CHAIMA','BU01UD09JE19MA97');
6  bitcoin.createNewBlock(2389,'OIUOEREDHKHKD','78s97d4x6dsf')
7  //bitcoin.createNewTransaction(100,'KAWTER-BOUDGHENE-STAMBOULI','CHAIMA-BOUDJEMAA');
8  //bitcoin.createNewTransaction(200,'NADJET-BOUCHENAFA','RIHAM-BOUDGHENE-STAMBOULI');
9  //bitcoin.createNewTransaction(300,'KAWTER-BOUDGHENE-STAMBOULI','MERIEM-BOUCIF');
10 //bitcoin.createNewTransaction(100000,'CHAIMA-OUALICHAOUCHE','KAWTER-BOUDGHENE-STAMBOULI');
11
12 //console.log(bitcoin.chain[2]);
13
14
15 console.log(bitcoin);
16
```

Figure IV- 60: Test de la méthode creatNewBlock

Maintenant, quand nous exécutez notre fichier test.js, nous devrions avoir trois blocs dans notre chaîne, comme le montre la capture d'écran suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
Blockchain {
  chain: [
    {
      index: 1,
      timestamp: 1586285086124,
      transactions: [],
      nonce: 1998,
      hash: 'BO14UD05GH19en98',
      previousBlockHash: 'KAWTER'
    },
    {
      index: 2,
      timestamp: 1586285086124,
      transactions: [],
      nonce: 1997,
      hash: 'BU01UD09JE19MA97',
      previousBlockHash: 'CHAIMA'
    },
    {
      index: 3,
      timestamp: 1586285086124,
      transactions: [],
      nonce: 2389,
      hash: '78s97d4x6dsf',
      previousBlockHash: 'OIUOEREDHKHKD'
    }
  ],
  newTransactions: [],
  newTransaction: []
}
```

Figure IV- 61: Exécution de la méthode creatNewBlock

IV.9.2.2 Créations des nouvelles transactions

La prochaine méthode que nous allons ajouter notre fonction constructrice de blockchain est appelée createNewTransaction.

Cette méthode créera une nouvelle transaction pour nous. Suivons ce qui suit étapes mentionnées pour créer la méthode :

```
23 Blockchain.prototype.getLastBlock = function () {
24     return this.chain[this.chain.length - 1];
25 }
26 Blockchain.prototype.createNewTransaction = function (amount, sender, recipient) {
27     const newTransaction = {
28         amount: amount,
29         sender: sender,
30         recipient: recipient,
31
32     };
33     this.newTransactions.push(newTransaction);
34     return this.getLastBlock()['index'] + 1;
35 }
```

Figure IV- 62: Programmations des nouvelles transactions

- *Tester d'ajoute des nouvelles transactions :*

Nous allons tester notre méthode createNewTransaction dans notre fichier test.js. Jetez un œil à la capture d'écran suivante pour un test rapide :

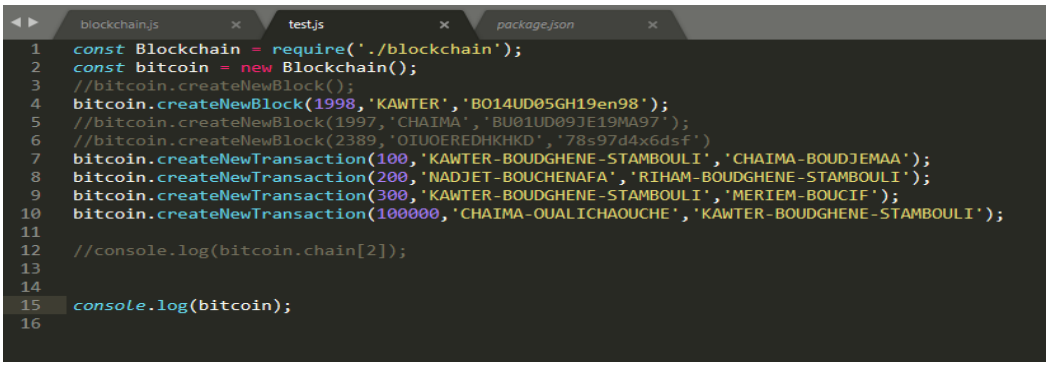


Figure IV- 63: Test d'ajoute des nouvelles transactions

Nous pouvons observer la blockchain bitcoin sur la fenêtre du terminal, comme le montre la capture d'écran suivante :

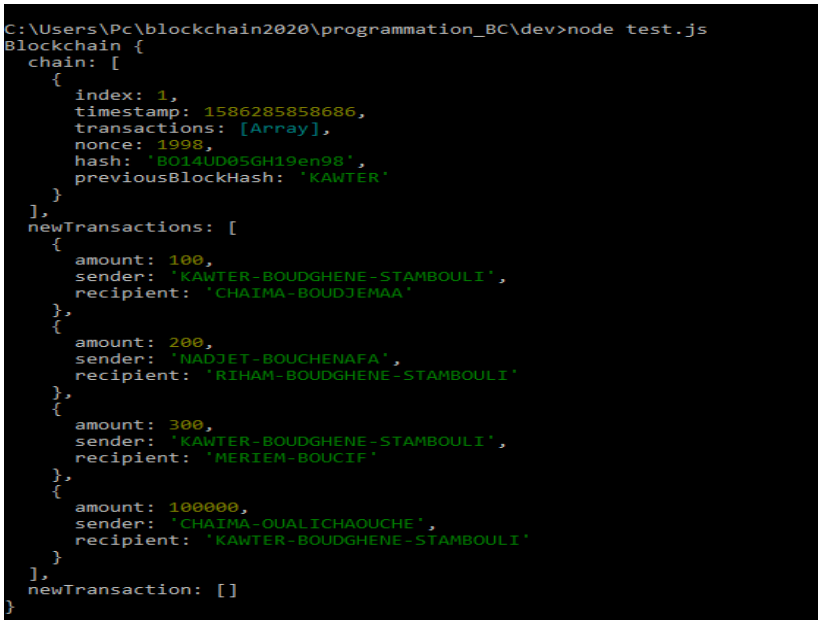


Figure IV- 64: Test d'ajoute des nouvelles transactions

IV.9.2.3 Mining d'un bloc

On va créer un bloc, créé une transaction, puis miné le nouveau bloc. Maintenant, la transaction nous avons créé devrait apparaître dans notre deuxième bloc, car nous avons miné un bloc après avoir créé une transaction comme le montre la figure suivante :

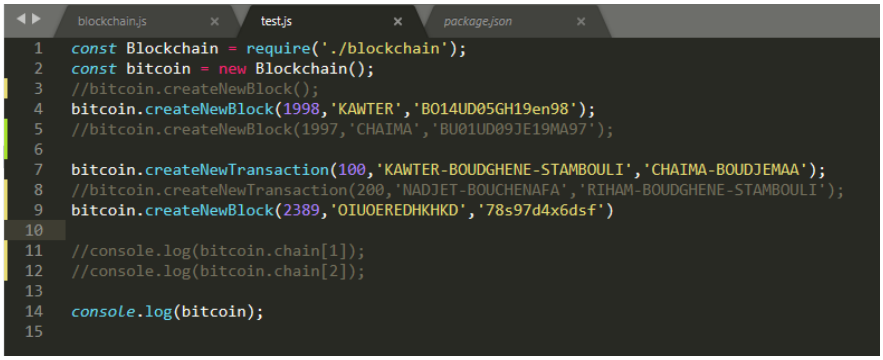


Figure IV- 65: Test de miner un bloc

- *Test de Mining un bloc :*

Voyons ce que nous donne :

Figure IV- 66: Exécution de minerde deux blocs

Nous avons à nouveau toute notre blockchain, qui contient deux blocs parce que nous avons miné deux blocs. Notre premier bloc (index : 1), qui n'a aucune transaction et à notre deuxième bloc (index : 2), dans lequel, si vous regardez nos transactions, il est dit qu'il y a un tableau qui contient des éléments par rapport à un tableau de transactions d'un premier bloc, qui ne contient aucun élément. Nous devons nous attendre à voir la transaction que nous avions créée précédemment.

En faisons la modification suivante à notre test:console.log(bitcoin.chain[1]);

Dans la sortie, vous pouvez voir que, pour les transactions, il contient un tableau avec un objet. Découvrons la capture d'écran suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
{
  index: 2,
  timestamp: 1586298024004,
  transactions: [
    {
      amount: 100,
      sender: 'KAWTER-BOUDGHENE-STAMBOULI',
      recipient: 'CHAIMA-BOUDJEMAA'
    }
  ],
  nonce: 2389,
  hash: '78s97d4x6dsf',
  previousBlockHash: 'OIUOEREDHKHKD'
}
```

Figure IV- 67: Modifications sur les transactions

Ce que nous avons fait ici était simplement de créer une transaction, puis de miner en créant un nouveau bloc ou en extrayant un nouveau bloc, qui contient maintenant notre transaction. Maintenant, réalisons quelques autres exemples pour aider à clarifier ce qui se passe ici.

```
bitcoin.createNewBlock(2389,'OIUOEREDHKHKD','78s97d4x6dsf')
bitcoin.createNewTransaction(200,'NADJET-BOUCHENAFA','RIHAM-BOUDGHENE-STAMBOULI');
bitcoin.createNewTransaction(300,'MARWA-BST','LOKMANE-BST');
bitcoin.createNewTransaction(800,'AMINA','KAWTER');
```

Figure IV- 68: Ajout des transactions successives

À ce stade, ces trois nouvelles transactions devraient se trouver dans notre tableau en attente de transactions, car nous ne créons pas de nouveau bloc après avoir créé ces trois transactions. Enfin, nous nous déconnectons à nouveau de notre blockchain bitcoin. Notre test devrait maintenant ressembler à ce qui suit :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
Blockchain {
  chain: [
    {
      index: 1,
      timestamp: 1586298909922,
      transactions: [],
      nonce: 1998,
      hash: 'BO14UD0SGH19en98',
      previousBlockHash: 'KAWTER'
    },
    {
      index: 2,
      timestamp: 1586298909922,
      transactions: [Array],
      nonce: 2389,
      hash: '78s97d4x6dsf',
      previousBlockHash: 'OIUOEREDHKHKD'
    }
  ],
  newTransactions: [],
  pendingTransactions: [
    {
      amount: 200,
      sender: 'NADJET-BOUCHENAFA',
      recipient: 'RIHAM-BOUDGHENE-STAMBOULI'
    },
    { amount: 300, sender: 'MARWA-BST', recipient: 'LOKMANE-BST' },
    { amount: 800, sender: 'AMINA', recipient: 'KAWTER' }
  ]
}
```

Figure IV- 69: Transactions restantes en attente

Dans la capture d'écran précédente, vous pouvez observer que nous avons notre blockchain. Dans cette chaîne, nous avons deux blocs, comme nous nous y attendions, et dans notre tableau de transactions en attente, nous avons trois transactions, qui sont les trois transactions que nous avons créées dans notre fichier de test. Ce que nous devons faire ensuite, c'est intégrer ces transactions en attente dans notre chaîne. Pour cela, minons un autre bloc. Copiez et collez simplement la méthode creatNewBlock après les trois transactions que nous avons créées et apportons des modifications à ses paramètres comme nous souhaitons. Lorsque nous exécutons le test maintenant, les trois transactions en

attente devraient apparaître dans notre nouveau bloc en faisant une petite modification de la dernière ligne de notre code de test, qui est console.log (bitcoin.chain [2]); la valeur 2 spécifie ici le troisième bloc du chaîne. Enregistrons le fichier et exécutons le test. Vous pourrez observer la sortie suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
{
  index: 3,
  timestamp: 1586299643174,
  transactions: [
    {
      amount: 200,
      sender: 'NADJET-BOUCHENAFA',
      recipient: 'RIHAM-BOUDGHENE-STAMBOULI'
    },
    { amount: 300, sender: 'MARWA-BST', recipient: 'LOKMANE-BST' },
    { amount: 800, sender: 'AMINA', recipient: 'KAWTER' }
  ],
  nonce: 1997,
  hash: 'BU01UD09JE19MA97',
  previousBlockHash: 'CHAIMA'
}
```

Figure IV- 70: Minage réussi

Vous pouvez voir que nous avons les trois transactions que nous avons créées. C'est ainsi que comment nos méthodes createNewTransactionset createNewBlock fonctionnent ensemble.

IV.9.3 Hachage des données

IV.9.3.1 Création de la méthode hashBlock

Construisons notre méthode hashBlock. À l'intérieur de cette méthode, nous voulons utiliser le hachage SHA256 pour hacher nos données de bloc. Suivez les étapes mentionnées dans la figure ci-dessous :

```
Blockchain.prototype.hashBlock = function(previousBlockHash, currentBlockData, nonce) {
    const dataAsString = previousBlockHash + nonce.toString()+ JSON.stringify(currentBlockData);
    const hash = sha256(dataAsString);
    return hash;
}
```

Figure IV- 71: Construction de la méthode hashBlock

Nous créons notre propre fonction hashBlock utilisée pour donner à chaque bloc son propre hachage. Comme vous pouvez le voir avec la fonction, le hachage prend chaque morceau de l'objet bloc, le jette dans une fonction SHA256 et le convertit en chaîne. Dans ce projet, nous convertissons les données, nonce et le bloc hash précédent.

C'est ainsi que notre méthode hashBlock fonctionnera. Dans la section suivante, nous allons tester la méthode pour voir si cela fonctionne parfaitement.

Test de la méthode hashblock :

Testons notre méthode hashBlock dans le fichier test.js. Semblable à ce que nous avons fait dans les sections précédentes, dans notre fichier test.js, nous devrions importer notre structure de données blockchain, créons une nouvelle instance de notre blockchain et la nommons bitcoin. Maintenant, testons notre méthode hashBlock.

Pour cela, saisissons la ligne de code suivante dans notre fichier test.js :

```
blockchain.js    test.js
1  const Blockchain = require('./blockchain');
2  const bitcoin = new Blockchain();
3
4  const previousBlockHash = '173359808ADFE657216AE69046E6E5ECF1214EDAB2D3E3AFDD0B05592AC26BBC';
5  const currentBlockData = [
6      {
7          amount : 10,
8          sender : 'KAWTER-BOUDGHENE-STAMBOULI',
9          recipient : 'CHAIMA-BOUDJEMAA',
10     },
11     {
12         amount : 30,
13         sender : 'NADJET-BOUCHENAFA',
14         recipient : 'RIHAM-BOUDGHENE-STAMBOULI',
15     },
16
17 ];
18 const nonce = 100;
19 console.log(bitcoin.hashBlock(previousBlockHash, currentBlockData, nonce));
20
```

Figure IV- 72: Test de la méthode hashBlock

Dans ce cas de test, nous appelons notre méthode hashBlock avec tous les paramètres corrects. Lorsque nous exécutons ce fichier, nous devons observer le hachage sur la fenêtre du terminal.

Maintenant, nous allons enregistrer ce fichier test.js et exécutons-le pour vérifier si nous obtenons ou non la sortie voulue. Nous allons accéder à la fenêtre de notre terminal et saisissons la commande node test.js, et observons les résultats obtenus. Nous pourrons observer le hachage résultant similaire en sortie de notre méthode hashBlock comme suit :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
internal/modules/cjs/loader.js:983
  throw err;
  ^

Error: Cannot find module 'sha256'
Require stack:
- C:\Users\Pc\blockchain2020\programmation_BC\dev\blockchain.js
- C:\Users\Pc\blockchain2020\programmation_BC\dev\test.js
[90m    at Function.Module._resolveFilename (internal/modules/cjs/loader.js:980:15)[39m
[90m    at Function.Module._load (internal/modules/cjs/loader.js:862:27)[39m
[90m    at Module.require (internal/modules/cjs/loader.js:1040:19)[39m
[90m    at require (internal/modules/cjs/helpers.js:72:18)[39m
    at Object.<anonymous> (C:\Users\Pc\blockchain2020\programmation_BC\dev\blockchain.js:1:16)
[90m    at Module._compile (internal/modules/cjs/loader.js:1151:30)[39m
[90m    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1171:10)[39m
[90m    at Module.load (internal/modules/cjs/loader.js:1000:32)[39m
[90m    at Function.Module._load (internal/modules/cjs/loader.js:899:14)[39m
[90m    at Module.require (internal/modules/cjs/loader.js:1040:19)[39m {
  code: [32m'MODULE_NOT_FOUND'[39m,
  requireStack: [
    [32m'C:\\Users\\Pc\\blockchain2020\\programmation_BC\\dev\\blockchain.js'[39m,
    [32m'C:\\Users\\Pc\\blockchain2020\\programmation_BC\\dev\\test.js'[39m
  ]
}
```

Figure IV- 73: Exécution de la méthode hashbBock

Nous avons oublié d'enregistrer la bibliothèque sha256. Nous devrons taper la commande suivante dans notre terminal : npm i sha256 --save

Nous verrons alors la sortie suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>npm i sha256 --save
npm WARN programmation_bc@1.0.0 No description
npm WARN programmation_bc@1.0.0 No repository field.

+ sha256@0.2.0
added 4 packages from 1 contributor and audited 4 packages in 4.703s
found 0 vulnerabilities
```

Figure IV- 74: Enregistrement de la bibliothèque sha256

Ne vous inquiétez pas de l'avertissement pour l'instant car cela ne nous affectera pas dans le cadre de ce projet. L'erreur existe car le dossier que nous avons créé est extrêmement dépouillé et ne contient pas de fichiers supplémentaires.

On va accédez à notre fenêtre de terminal et tapons la commande node test.js, et observons les résultats que nous obtenons. Normalement, maintenant, on pourra observer le hachage résultant en sortie :

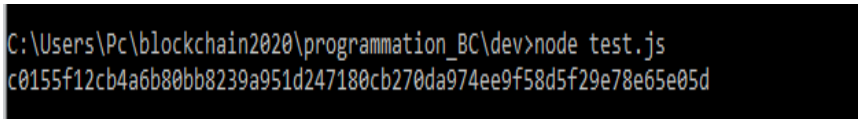


Figure IV- 75: Résultat de hachage

Il semble que notre méthode hashBlock fonctionne bien.

IV.9.4 Proof of Work

IV.9.4.1 **Définition**

La prochaine méthode que nous allons ajouter à notre structure de données blockchain est la méthode de preuve de travail. Cette méthode est très importante et essentielle à la technologie blockchain.

La méthode ProofOfWork prendra les deux paramètres : currentBlockData et previousBlockHash. À partir de ces données que nous fournissons, la méthode proofOfWork tentera de générer un hachage spécifique. Ce hachage spécifique dans notre exemple va être un hachage qui commence par quatre zéros.

Essayons maintenant de comprendre comment nous pouvons le faire. Comme nous l'avons appris dans les sections précédentes, le hachage généré à partir de SHA256 est à peu près aléatoire. Donc, si le hachage résultant est à peu près aléatoire, comment pouvons-nous générer un hachage à partir de notre bloc actuel qui commence par quatre zéros ? La seule façon de le faire est par essais et erreurs, ou en devinant et en vérifiant. Donc, cette fois, nous allons incrémenter notre valeur nonce de 1. Si nous n'obtenons pas à nouveau la valeur de hachage correcte, nous allons incrémenter la valeur nonce et réessayer. Si cela ne fonctionne pas, nous incrémentons à nouveau la valeur nonce et réessayons. Ensuite, nous exécuterons continuellement cette méthode hashBlock jusqu'à ce que nous trouvions un hachage qui commence par quatre zéros. C'est ainsi que notre méthode proofOfWork fonctionnera. Donc, si quelqu'un voulait retourner dans la blockchain et essayer de changer un bloc ou les données de ce bloc - peut-être pour se donner plus de Bitcoin - il faudrait faire une tonne de calculs et utiliser beaucoup d'énergie pour créer le bon hacher. Dans la plupart des cas, il n'est pas possible de revenir en arrière et d'essayer de recréer un bloc déjà existant ou de réexploiter un bloc déjà existant avec vos propres faux donnés. En plus de cela, notre méthode hashBlock prend non seulement le currentBlockData, mais aussi le précédent BlockHash. Cela signifie que tous les blocs de la blockchain sont liés entre eux par leurs données. Cela peut sembler écrasant un peu pour le moment, mais ne vous inquiétez pas - nous allons construire la méthode ProofOfWork ci-dessous, puis nous allons la tester avec de nombreux types de données différents. Cela nous aidera à nous familiariser avec le fonctionnement de la méthode ProofOfWork et la manière dont elle sécurise la blockchain.

IV.9.4.1 Création de la méthode Proof of Work

Alors, on commence par définir la méthode ProofOfWork :

```
Blockchain.prototype.proofOfWork = function(previousBlockHash, currentBlockData) {
    let nonce = 0;
    let hash = this.hashBlock(previousBlockHash, currentBlockData, nonce);
    while (hash.substring(0, 4) !=='0000') {
        nonce++;
        hash = this.hashBlock(previousBlockHash, currentBlockData, nonce);
        //console.log(hash);
    }
    return nonce;
}
```

Figure IV- 76: Configuration de la preuve de travail

C'est ainsi que notre méthode proofOfWork fonctionnera et validera le hachage.

Dans la section suivante, nous testerons notre méthode proofOfWork dans notre fichier test.jspour nous assurer qu'elle fonctionne correctement. Nous étudierons également pourquoi nous retournons une valeur nonce au lieu de renvoyer le hachage.

- *Teste de la méthode proofOfWork :*

Pour tester notre méthode proofOfWork, nous avons besoin de previousBlockHash et currentBlockData. Donc, dans notre cas de test, supprimons la valeur nonce et ajoutons les lignes de code suivantes à notre fichier :

console.log(bitcoin.proofOfWork(previousBlockHash, currentBlockData));

Une fois le test exécuté, nous remarquerons qu'un nombre apparaît en sortie à l'écran :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
61989
```

Figure IV- 77: Obtention du nonce

Ce que signifie ce nombre, c'est qu'il a fallu 61989 itérations pour que notre méthode proofOfWork trouve un hachage qui commence par quatre zéros.

Nous devrons apporter des modifications mineures à notre boucle while, ce qui va se passer, c'est que nous devrions réellement voire plus de 61000 hachages différents affichés dans notre terminal. Aucun de ces hachages ne commencera par quatre zéros, à l'exception du dernier. Seul le dernier hachage affiché commence avec quatre zéros car après notre méthode, cela se terminera et renverra la valeur nonce pour laquelle le hachage valide a été obtenu.

Nous pouvons maintenant observer sur notre écran que nous avons toute une tonne de hachages différents enregistrés sur terminal :

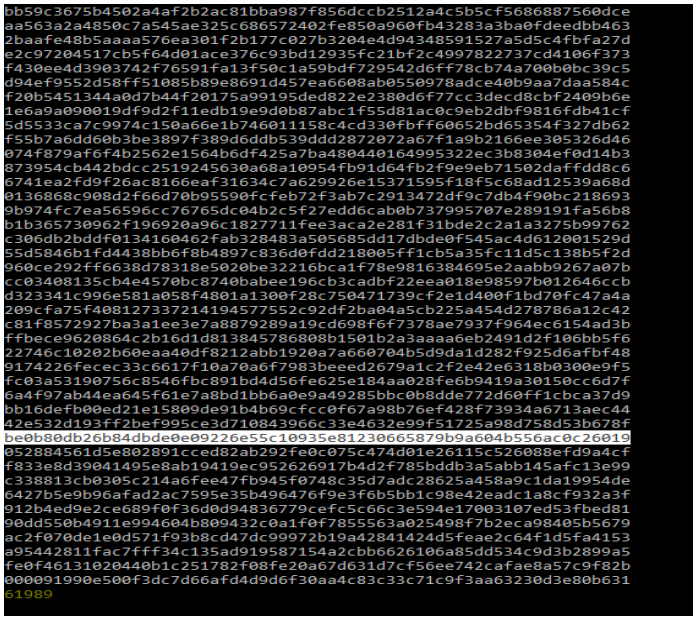


Figure IV- 78: Exécution de PoW

Nous pouvons également observer que pour chaque hachage qui a été affiché, le début n'est jamais quatre zéros de suite jusqu'à ce que nous obtenions notre valeur finale.

Essayons maintenant d'utiliser notre méthode hashBlock. Dans notre fichier test.js, supprimons la méthode proofOfWork et ajoutons la ligne de code suivante :

```
console.log(bitcoin.hashBlock(previousBlockHash, currentBlockData, 61989));
```

En faisant cela, nous devons générer un hachage qui commence par quatre zéros au premier essai. Enregistrons-le et exécutons-le. Une fois le test exécuté, nous pourrons observer le hachage unique qui commence par quatre zéros, comme indiqué dans la capture d'écran suivante :

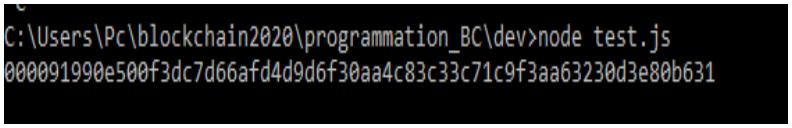


Figure IV- 79: Vérification du hash après l'obtention du nonce

Ainsi, à partir de notre test, nous pouvons conclure que la méthode proofOfWork fonctionne comme prévu.

IV.9.4 Création du bloc genesis :

Une autre chose que nous devrions ajouter à notre structure de données blockchain est le bloc genesis. Mais qu'est-ce qu'un bloc de genèse ? Eh bien, un bloc de genèse est simplement le premier bloc d'une chaîne de blocs.

À l'intérieur de la fonction constructeur blockchain tapons la ligne de code suivante : this.createNewBlock(1405, '0', '0');

Dans le code précédent, nous avons passé la valeur nonce à 1405, previousBlockHash à 0 et la valeur de hachage à 0. Ce ne sont que des valeurs arbitraires ; vous pouvez ajouter la

valeur que vous souhaitez ajouter. Maintenant, enregistrons le fichier et testons le bloc genesis dans le fichier test.js. Comme le montre la capture d'écran suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node test.js
Blockchain {
  chain: [
    {
      index: 1,
      timestamp: 1586461441725,
      transactions: [],
      nonce: 1405,
      hash: '0',
      previousBlockHash: '0'
    }
  ],
  newTransactions: [],
  pendingTransactions: []
}
```

Figure IV- 80: Créations du bloc genesis

Dans la capture d'écran précédente, pour le tableau de chaînes, on peut voir que nous avons un bloc à l'intérieur de la chaîne. Ce bloc est notre bloc de genèse et il a un nonce de 1405, un hachage de 0 et un précédent Blockhash de 0. Dorénavant, toutes nos chaînes de blocs auront un bloc de genèse.

IV.8.6 Création d'une API Express

Dans cette section, nous allons créer une API de liste de tâches RESTful (c'est-à-dire des points de terminaison qui créeront une tâche, obtiendront ou liront la liste de toutes les tâches, liront une tâche particulière, supprimeront une tâche et mettront à jour une tâche). À l'aide d'Express et couvrir quelques bonnes pratiques de développement. On découvre comment toutes les pièces fonctionnent ensemble lorsqu'on crée un projet.

IV.8.6.1 Configuration du serveur

Installons express et nodmon, express sera utilisé pour créer le serveur tandis que nodmon nous aidera à garder une trace des modifications de notre application en regardant les fichiers modifiés et redémarrer automatiquement le serveur.

IV.8.6.1.1 Installation d'express

Nous pouvons maintenant installer Express on exécute cette commande :

```
#npm i express
```

IV.8.6.1.2 Crée le fichier serveur.js

Au début du code est importé le express module et créé une app. Après avoir créé un itinéraire basé sur la méthode HTTP. Les objets reset req sont fournis par le nœud, ce qui équivaut à une requête HTTP et à une réponse. Pour terminer, je dis à l'instance créée d'écouter sur le port 3000

```
1  var express = require('express');
2  var app = express();
3  app.get('/', function (req, res) {
4  res.send('Hello world');
5  });
6  app.listen(3000);
```

Figure IV- 81: Création de fichier serveur.js

IV.8.6.1.3 Démarrer le serveur *express*

Si tout s'est bien passé, rendez-vous sur localhost : 3000

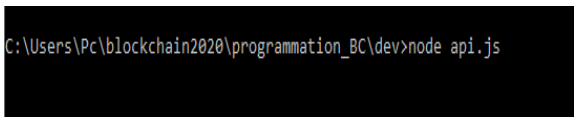


Figure IV- 82: Démarrage du serveur express

Passons à chrome et allons au port localhost 3000, voici notre 'Hello Word '

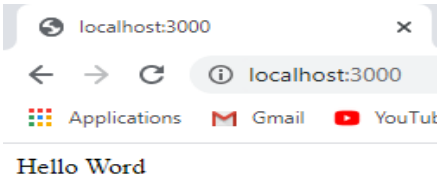


Figure IV- 83: Exécution du serveur

Ce fichier est aussi simple que possible lors de la définition d'une application Express. Nous avons créé un objet d'application et l'avons exporté pour utilisation.

IV.8.6.1.4 Configuration des itinéraires

Le routage consiste à déterminer comment une application répond à une demande client pour un point de terminaison spécifique, qui est un URI (ou chemin) et une méthode de demande HTTP spécifique (GET, POST, etc.).

- L'URI (ou chemin) de la demande que nous voulons capturer.

```
app.get ('/') ;
```

- Une fonction de gestionnaire (qui accepte un objet de demande et de réponse comme arguments).

```
app.get ('/' , fonction (req, res) {} );
```

IV.8.6.1.5 Quelle méthode HTTP devrions-nous utiliser ?

Lors de la construction d'une API REST, chaque méthode HTTP correspond à une action contre une ressource servie par l'API.

- *GET :* récupérer l'objet d'une ressource particulière ou répertorier tous les objets
- *POST :* créer un nouvel objet de ressource
- *PATCH :* effectuer une mise à jour partielle de l'objet d'une ressource particulière
- *PUT :* écrase complètement l'objet d'une ressource particulière
- *DELETE :* supprime l'objet d'une ressource particulière

Nous pouvons commencer à définir notre API en spécifiant les itinéraires pour une seule ressource.

IV.8.6.2 *Construire les fondations d'API*

Dans cette section, nous allons continuer à construire notre API blockchain. Puis nous allons commencer par construire les trois points de terminaison suivants dans notre API :

- Le premier point de terminaison est / blockchain, ce qui nous permet d'aller chercher notre blockchain entière afin que nous puissions regarder les données qui s'y trouvent.
- Le deuxième point de terminaison est / transaction, qui nous permet de créer une nouvelle transaction.
- Le troisième critère est / le mine, qui nous permettent d'exploiter un nouveau bloc en utilisant la méthode ProofOfWork que nous avons faite dans la dernière section.

Dans le fichier api.js, définissons ces points de terminaison comme suit :

```
1
2   const express = require('express');
3   const app = express();
4   app.get('/', function (req, res) {
5       res.send('Hello World');
6   });
7   app.get('/blockchain', function (req, res) {
8   });
9   app.post('/transaction', function(req, res) {
10  });
11  app.get('/mine', function(req, res) {
12  });
13  app.listen(3000);
```

Figure IV- 84: Interface d'applications Blockchain

Maintenant, une autre chose que nous voulons faire est apportez quelques modifications à la méthode listen :

```
app.listen(3000, function(){console.log('listening on port
3000...') ;
```

À l'intérieur de cette fonction, nous allons simplement imprimer l'écoute sur la chaîne du port 3000. La raison pour laquelle nous faire est juste pour que lorsque notre port est en cours d'exécution, nous verrons ce texte. Allons allez sur notre terminal et on lance notre fichier api.js encore :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>node api.js
listening on port 3000...
```

Comme nous pouvons le voir, la capture d'écran précédente nous montre que nous écoutons le port 3000. Chaque fois que nous voyons ce texte, nous savons que, nos serveurs fonctionnent.

IV.8.6.2.1 Installation de Postman et de l'analyseur de corps

La première chose qui nous va faire est d'installer un nouveau package appeler nodemon. Dans notre répertoire programmation_BC dans notre terminal, nous écrirons la commande suivante :

```
#npm i nodemon --save
```

Après cela, nous apportons une modification dans l'un de nos fichiers et l'enregistrons. Cette bibliothèque nodemon redémarrer automatiquement notre serveur pour nous, afin que nous ne devions pas aller et venir du terminal à notre code pour redémarrer le serveur chaque fois que nous faisons un changement.

Donc, nous allons ouvrir notre fichier package.json. Où il est écrit "scripts", nous allons ajouter un nouveau script :

```
{
  "name": "programmation_bc",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
    "start": "nodemon --watch dev -e js dev/api.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.1",
    "nodemon": "^2.0.3",
    "sha256": "^0.2.0"
  }
}
```

Figure IV- 85: Automatisation de ce processus

Chaque fois que l'un de ces fichiers JS sont modifiés et enregistrés, nous voulons nodemon pour redémarrer notre fichier dev/api.js pour nous. Sauvegarder le fichier package.json. Maintenant, chaque fois que nous faisons un changement à l'intérieur de notre dossier de développement et l'enregistrer, le serveur se redémarrera tous seul. Testons cela.

Allons à notre terminal. Notre serveur devrait maintenant utiliser nodemon après cette commande

```
C:\Users\Pc\blockchain2020\programmation_BC>npm start

> programmation_bc@1.0.0 start C:\Users\Pc\blockchain2020\programmation_BC
> nodemon --watch dev -e js dev/api.js

[nodemon] 2.0.3
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): dev\**\*
[nodemon] watching extensions: js
[nodemon] starting `node dev/api.js`
listening on port 3000...
```

Figure IV- 86: Exécution de nodemon

Nous verrons que notre serveur redémarre tous seul, on va faire une petite modification sur le texte Hello world

```
app.get('/', function (req, res) {res.send('Hello World!!!');
});
```

Ceci est juste un outil que nous utilisons pour rendre le développement un peu plus facile pour nous.

Maintenant, un autre outil que nous voulons utiliser est appelé Postman. L'outil Postman nous permet d'appeler n'importe quels points de terminaison, ainsi que d'envoyer des données à ces points de terminaison avec nos demandes.

IV.8.6.2.2 L'utilisation de Postman

Téléchargez l'application. Une fois que nous avons téléchargé l'application, nous pouvons effectuer un petit essai de la façon dont nous pouvons utiliser cette application Postman pour atteindre notre point de terminaison / transaction.

Pour tester que le point de terminaison / transaction est travailler, renvoyons quelque chose le résultat. Dans notre point de terminaison / transaction, nous avons ajouté la ligne suivante

```
app.post('/transaction', function(req,res) {res.send('It works!!!');
```

Et maintenant, quand nous atteignons ce point, nous devrions obtenir le texte It works!!! Cliquons sur Envoyer et nous obtiendrons la sortie, comme indiqué dans la capture d'écran suivante :

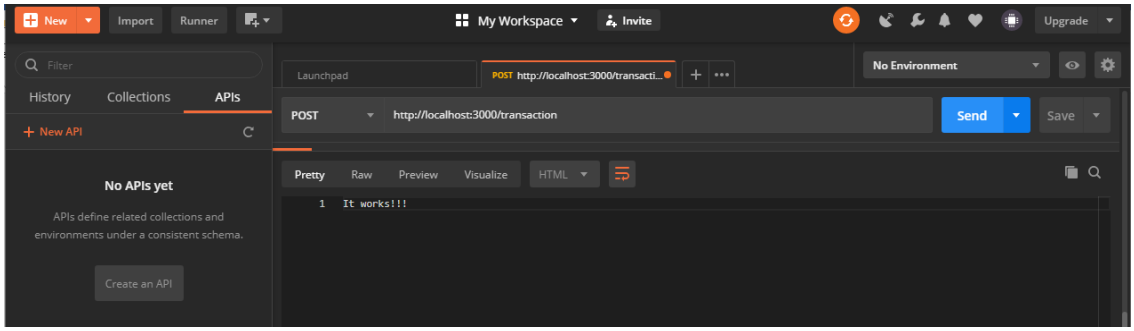


Figure IV- 87: Tester le point de terminaison / transaction

Maintenant, la plupart du temps, nous frappons un message dans notre API, nous allons vouloir lui envoyer des données. Par exemple, lorsque nous atteignons le endpoint / transaction, nous souhaitons créer une nouvelle transaction. Par conséquent, nous devons envoyer les données de transaction, telles que le montant de la transaction, l'expéditeur et le destinataire. Nous pouvons le faire en utilisant Postman, et c'est plus simple. Ce que nous allons faire ici, c'est envoyer des informations dans le corps de notre demande de publication. Vous pouvez le faire en cliquant sur l'onglet Corps comme le montre la figure suivante :

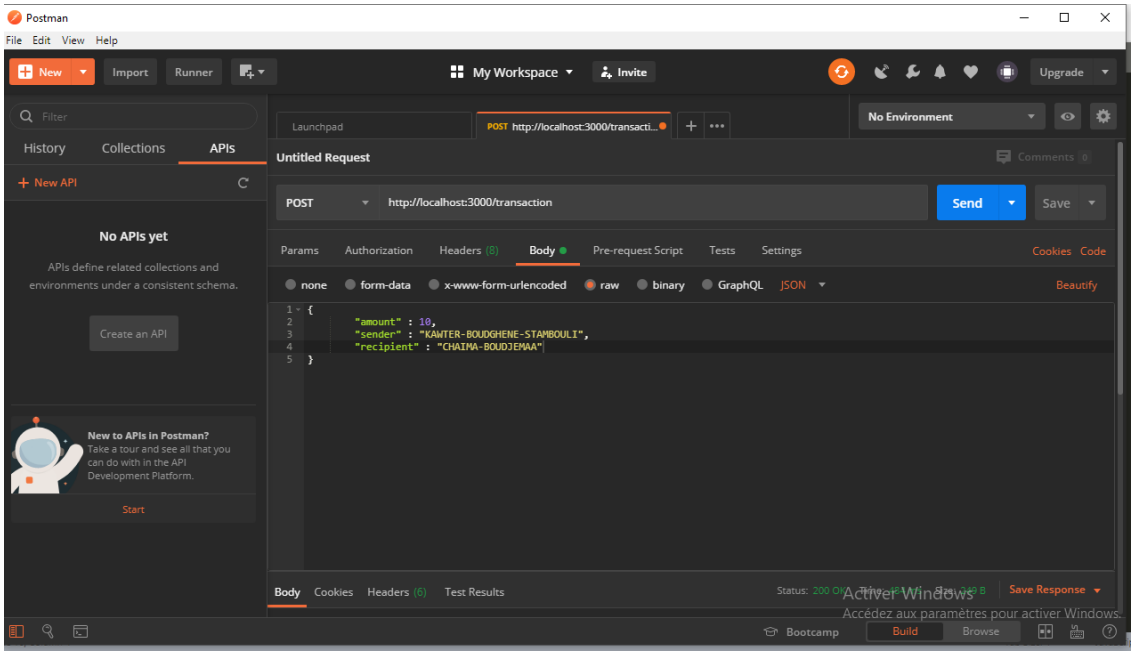


Figure IV- 88: Envoi des données de transaction à partir du postman

Nous pouvons également voir que nous avons fait un JSON objet et y mettre des données. On a ajouté le montant comme 10 bitcoins, le l'adresse de l'expéditeur et l'adresse du destinataire.

Pour tester si nous recevons ou non toutes ces informations à l'intérieur de notre endpoint, nous allons écrire l'ensemble du corps de la demande. Le req.body est simplement l'information que nous avons créée dans l'objet JSON:

```
app.post('/transaction', function(req, res) {
    console.log(req.body);
    res.send(`The amount of the transaction is ${req.body.amount} bitcoin.`);
});
```

Figure IV- 89: Configuration de req.body

Maintenant, pour que $ {req.body.amount} fonctionne, nous besoin d'installer une autre bibliothèque afin de accéder à ces informations. Revenons en notre terminal ; nous allons quitter le processus et nous allons installer un package appeler body-parser :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>npm i body-parser --save
npm WARN saveError ENOENT: no such file or directory, open 'C:\Users\Pc\blockchain2020\programmation_BC\dev\package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@~2.1.2 (node_modules\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.1.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN enoent ENOENT: no such file or directory, open 'C:\Users\Pc\blockchain2020\programmation_BC\dev\package.json'
npm WARN dev No description
npm WARN dev No repository field.
npm WARN dev No README data
npm WARN dev No license field.

+ body-parser@1.19.0
updated 1 package and audited 975 packages in 42.436s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Figure IV- 90: Installation de body-parser

Maintenant, redémarrons notre serveur avec npm start. Lorsqu'il s'agit d'utiliser l'analyseur de corps, nous voulons simplement l'importer en haut de notre fichier après la ligne où nous application importée.

```
const express = require('express');
const app = express();
const bodyParser = require('body-parser');

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: false }
```

Figure IV- 91: Importation de fichier body-parser

Maintenant que nous utilisons l'analyseur corporel, nous devrait pouvoir accéder au montant. Enregistrons le fichier api.js et essayons d'envoyer la demande.

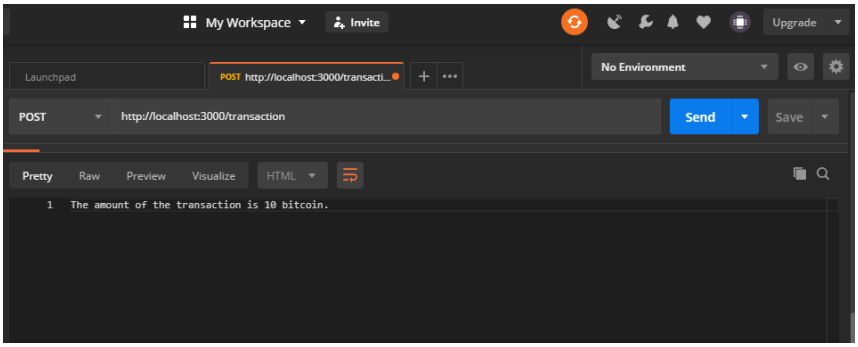


Figure IV- 92: Envoi de la demande

Ça a marché ! Nous avons renvoyé la chaîne, qui indique que le montant de la transaction est 10 bitcoins. Dans notre terminal, nous pouvons voir que l'ensemble informations concernant le montant, l'expéditeur, et le destinataire s'affiche :

```
C:\Users\Pc\blockchain2020\programmation_BC>npm start

> programmation_bc@1.0.0 start C:\Users\Pc\blockchain2020\programmation_BC
> nodemon --watch dev -e js dev/api.js

[nodemon] 2.0.3
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): dev\**\*
[nodemon] watching extensions: js
[nodemon] starting `node dev/api.js`
listening on port 3000...
{
  amount: 10,
  sender: 'KAWTER-BOUDGHENE-STAMBOULI',
  recipient: 'CHAIMA-BOUDJEMAA'
}
```

Figure IV- 93: Réception de la chaine envoyée

IV.8.6.2.3 Construire le /blockchain endpoint

Dans cette section, nous allons interagir avec notre endpoint / blockchain. Cela signifie que nous devra importer notre blockchain de notre Fichier blockchain.js comme ça :

```
const Blockchain = require('./blockchain');
const bitcoin = new Blockchain();
```

Figure IV- 94: Importation de fichier blockchain.js

Tout ce que ce endpoint va faire est de renvoyer toute notre blockchain à celui qui a appelé ce point de terminaison. Pour ce faire, nous allons ajouter une ligne qui enverra la réponse :

```
app.get('/blockchain', function(req, res) {
    res.send(bitcoin);
});
```

Figure IV- 95: Renvoie de la blockchain

Croyez-le ou non, c'est tout ce que nous allons faire pour ce endpoint.

- *Test de /blockchain endpoint :*

Maintenant, nous pouvons tester si ce endpoint fonctionne en l'utilisant dans notre navigateur :

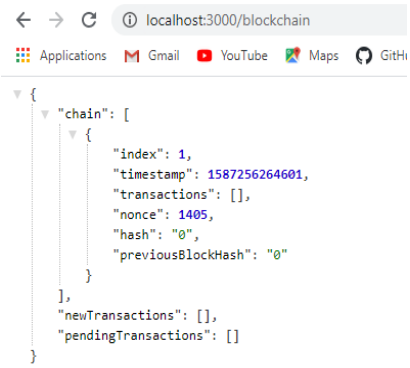


Figure IV- 96: Récupération de la chaine blockchain

Comme nous pouvons le voir, nous récupérons notre blockchain.

IV.8.6.2.4 Construire la /transaction endpoint

Dans notre / transaction endpoint, nous allons ajouter la ligne suivante :

```
app.post('/transaction', function(req, res) {
    const blockIndex = bitcoin.createNewTransaction(req.body.amount, req.body.sender, req.body.recipient
    res.json({ note:`Transaction will be added in block ${blockIndex}.`});
});
```

Figure IV- 97: Créer un nouvel objet de ressource s'appelle transaction

Le résultat sera enregistré dans blockIndex, et c'est ce que nous allons renvoyer à celui qui appelle ce point de terminaison. Nous vous le renverrons sous forme de note : Le résultat sera enregistré dans blockIndex, et c'est ce que nous allons renvoyer à celui qui appelle ce point de terminaison. Nous vous le renverrons sous forme de note.

Comme vous pouvez le voir, la note nous indiquera à quel bloc la transaction sera ajoutée. Nous avons utilisé l'interpolation de chaîne pour passer le valeur blockIndex. Enregistrons ce fichier et testons ce point de terminaison à l'aide de Postman.

- *Test de /transaction endpoint :*

Passons maintenant à Postman et appliquons des paramètres similaires à ceux que nous avons définis précédemment :

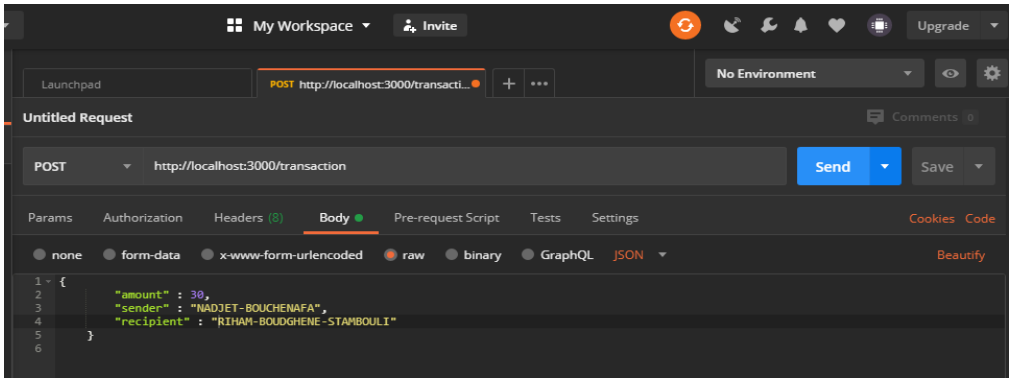


Figure IV- 98: Création des données

Testons ce point de terminaison :

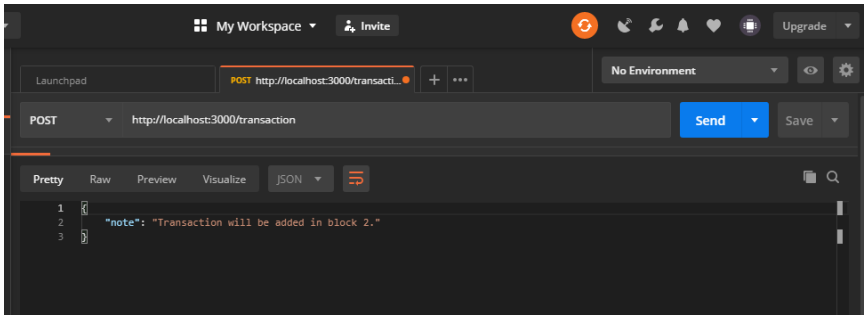


Figure IV- 99: Test du point de terminaison /transaction

Comme nous pouvons le voir, lorsque nous avons cliqué sur Envoyer bouton sur Postman, nous avons obtenu la sortie La transaction sera ajoutée dans le bloc 2. La raison pour laquelle nous avons le bloc 2 ici est qu'un bloc avait déjà été créé lorsque nous avons lancé notre blockchain, qui a créé la genèse bloquer. Par conséquent, cette transaction a ajouté au bloc 2.

Une autre façon que nous pouvons tester pour nous assurer que ce point de terminaison a fonctionné correctement est en appuyant sur notre / blockchain endpoint.

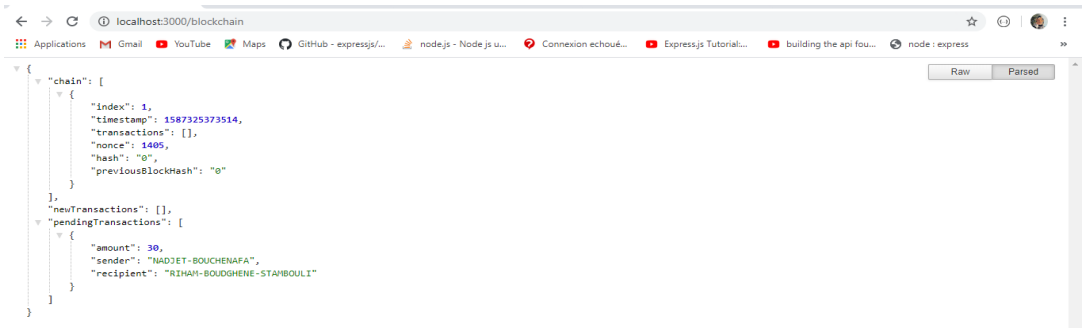


Figure IV- 100: Réception de la chaine dans le navigateur

Comme nous pouvons le voir, l'objet entier est notre tout blockchain - la première partie est notre chaîne qui a le bloc de genèse, et la deuxième partie est notre transaction en attente, que nous venons d'établi. Notre / transaction endpoint fonctionne à la perfection.

IV.8.6.2.5 Construire /mine endpoint

Construisons le point final pour notre API blockchain : le point final de la mine, cela minez et créez un nouveau bloc :

```
app.get('/mine', function(req, res) {
    const lastBlock = bitcoin.getLastBlock();
     const previousBlockHash = lastBlock['hash'];
     const currentBlockData = { transactions: bitcoin.pendingTransactions, index: lastBlock['index'] + 1 };
        const nonce = bitcoin.proofOfWork(previousBlockHash, currentBlockData);
        const blockHash = bitcoin.hashBlock(previousBlockHash, currentBlockData, nonce);
        const newBlock = bitcoin.createNewBlock(nonce, previousBlockHash, blockHash);
        bitcoin.createNewTransaction({
                        amount: 12.5,
                        sender: "00",
                        recipient: nodeAddress

                });
    res.json({note: "New block mined successfully",  block: newBlock});
});
```

Figure IV- 101: Construire /mine endpoint

Ce qui se passe ici est joli impressionnant. Comme nous pouvons le voir, il y a beaucoup de calculs différents qui entrent dans la création de ce nouveau bloc, et nous sommes en mesure de faire tous ces calculs en utilisant notre structure de données blockchain. Il s'agit d'une structure de données assez puissante, et notre blockchain peut désormais exploiter des nouveaux blocs en utilisant proofOfWork.

Nous allons également faire de notre récompense 12,5 bitcoins. En tant qu'adresse d'expéditeur, nous avons mis la valeur 00. De cette façon, chaque fois nous examinons les transactions sur notre réseau, nous savons que si une transaction est fabriquée à partir de l'adresse 00, c'est une récompense minière. Maintenant, tout ce dont nous avons besoin est l'adresse d'un destinataire, nodeAddress. Nous devons envoyer 12,5 bitcoins à celui qui a extrait un nouveau bloc - mais comment on trouve ça ? Eh bien, nous allons envoyer cette récompense au nœud actuel que nous sommes sur, qui est tout ce fichier API que nous sommes travaillé sur. Nous pouvons traiter cette API entière comme nœud de réseau dans la blockchain bitcoin.

Pour créer une adresse pour ce nœud, nous sommes va importer une nouvelle bibliothèque appelée uuid en utilisant notre terminal :

```
#npm i uuid
```

Importons maintenant notre nouvelle bibliothèque uuid aux sections supérieures de notre fichier api.js :

```
const { v1: uuidv1 } = require('uuid');
const nodeAddress = uuidv1().split('-').join('');
```

Figure IV- 102: Importation de la bibliothèque UUID

Cette bibliothèque crée une chaîne aléatoire pour nous, et nous allons utiliser cette chaîne comme adresse de ce nœud de réseau. NodeAddress que nous allons obtenir est aléatoire chaîne qui est garantie d'être unique, car nous ne voulons pas avoir deux nœuds avec la même adresse.

Nous testerons maintenant notre / mine endpoint, ainsi que notre / transaction et / blockchain, pour vous assurer qu'ils tous fonctionnent et interagissent correctement.

- *Test de /mine endpoint :*

Ouvrons un onglet et frappons notre / mine endpoint. Cela devrait exploiter et créer un nouveau bloqué pour nous :

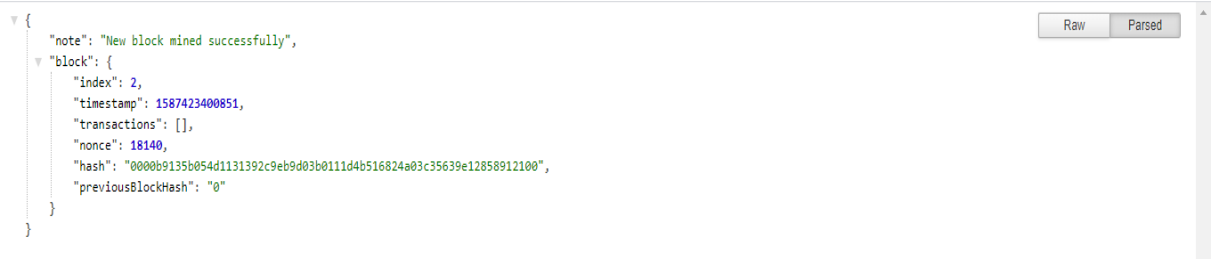


Figure IV- 103: Test de /mine endpoint

Nous avons obtenu notre note qui dit que le nouveau bloc a été extrait avec succès. Nous avons également récupéré notre nouveau bloc et nous pouvons voir toutes les données qui se trouvent sur notre bloc. Il contient un hachage, et il a également le hachage du bloc précédent, qui est le bloc de genèse, et une transaction en elle. Vous pourriez penser, nous n'avons pas créé une transaction, alors d'où vient cette transaction viens d'où ? Cette transaction est en fait la récompense minière que nous mettons dans notre endpoint, qui a la récompense minière de 12,5 bitcoins transaction. Il ressemble à notre point d'extrémité minier a bien fonctionné.

- *Test de /blockchain endpoint :*

Pour tester et nous assurer que nous avons créé ce nouveau bloc, nous pouvons retourner à notre / blockchain endpoint et actualisons la page :

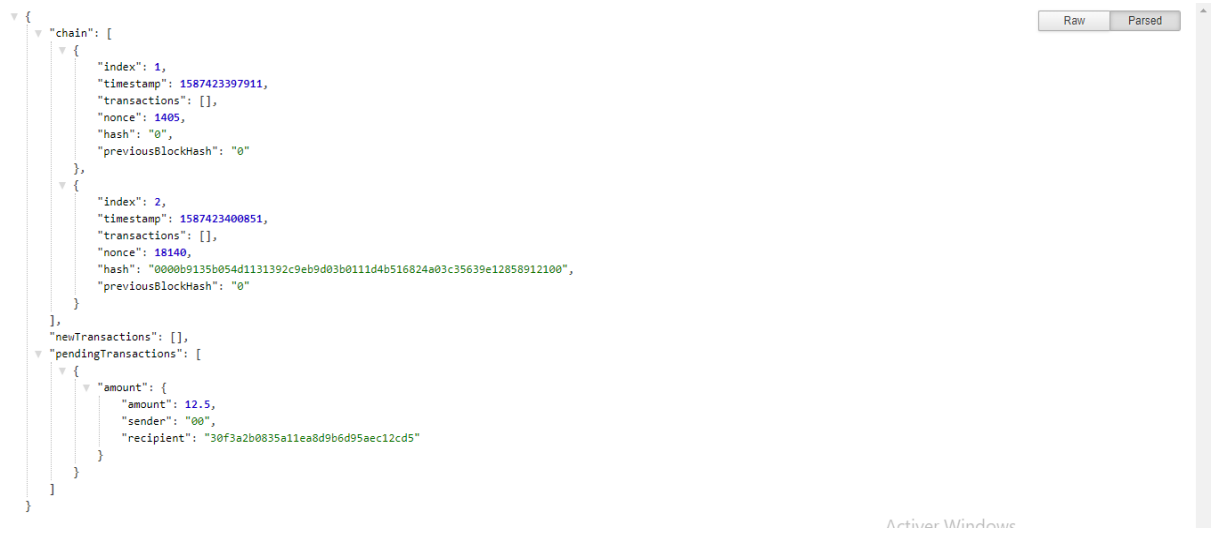


Figure IV- 104: Test de /blockchain endpoint

Ça a marché. Nous avons maintenant deux blocs dans notre chaîne : l'un est le bloc de genèse et l'autre est celui que nous venons de créer. Le deuxième bloc a également la transaction en elle.

Minons des autres blocs pour tester cela à nouveau. Rendez-vous sur notre endpoint / mine et actualisez la page à nouveau :

Figure IV- 105: Miner plusieurs blocs

Comme nous le voyez, nous avons les trois blocs. Le bloc 3 est celui que nous venons de créer, et il contient notre transaction de récompense minière. Une plus de chose à noter est que notre précédent BlockHash s'aligne réellement avec le hachage de notre bloc 2. C'est aider à sécuriser notre blockchain.

- *Test de /transaction endpoint :*

Créons maintenant quelques transactions avec notre / point de terminaison de transaction. Pour cela, rendez-vous sur Postman, assurez-vous que les paramètres sont les comme précédemment, et procédez comme suit changements :

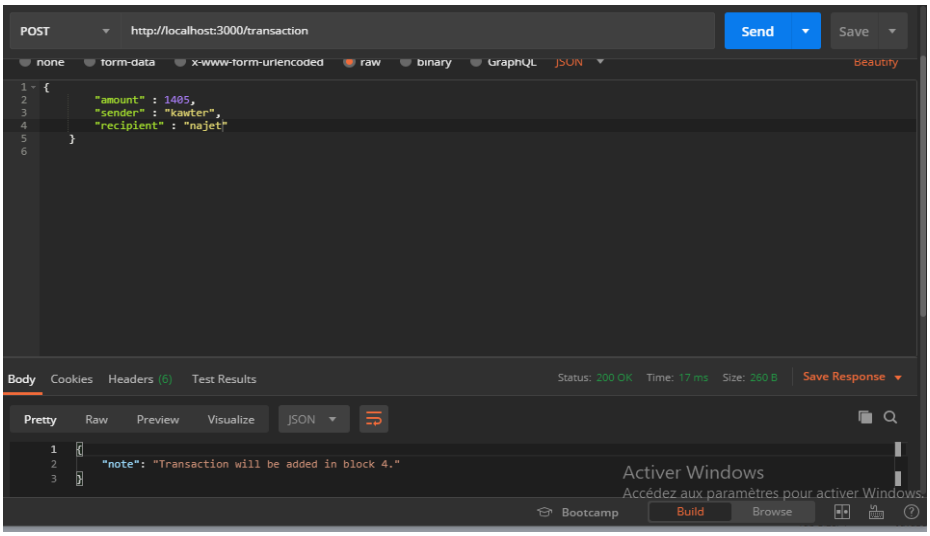


Figure IV- 106: Test de /transaction endpoint

Cette transaction a été ajoutée à bloc 4 parce que nous avons déjà trois blocs dans notre chaîne. Maintenant, allons-y et récupérons notre toute la blockchain à nouveau. Cette fois, nous devrions attendre à obtenir la même blockchain et deux transactions en attente que nous venons d'établi. Rafraîchissons la page et voyons la production :

```
  "newTransactions": [],
  "pendingTransactions": [
    {
      "amount": {
        "amount": 12.5,
        "sender": "00",
        "recipient": "30f3a2b0835a11ea8d9b6d95aec12cd5"
      }
    },
    {
      "amount": 1405,
      "sender": "NADJET-BOUCHENAFA",
      "recipient": "RIHAM-BOUDGHENE-STAMBOULI"
    },
    {
      "amount": 1405,
      "sender": "kawter",
      "recipient": "najet"
    }
  ]
}
```


Figure IV- 107: Exécution du Test de /transaction endpoint

On remarque que celui-ci comporte trois blocs et trois transactions en attente. Maintenant, si nous nous dirigeons vers notre point d'extrémité / mine et actualiser le page, ces trois transactions en attente seront ajoutées au bloc 4 :


```
{
  "note": "New block mined successfully",
  "block": {
    "index": 4,
    "timestamp": 1587424526972,
    "transactions": [
      {
        "amount": {
          "amount": 12.5,
          "sender": "00",
          "recipient": "30f3a2b0835a11ea8d9b6d95aec12cd5"
        }
      },
      {
        "amount": 1405,
        "sender": "NADJET-BOUCHENAFA",
        "recipient": "RIHAM-BOUDGHENE-STAMBOULI"
      },
      {
        "amount": 1405,
        "sender": "kawter",
        "recipient": "najet"
      }
    ],
    "nonce": 25275,
    "hash": "00003c6c1f0b8e382aad7f94f6d83ab7e6f2db9428682e714b671dec21ea7095",
    "previousBlockHash": "0000642946e5bc28bef7e55c1a6859590f663fcbb5e1ab40bd76e6b461ed783b"
  }
}
```


Figure IV- 108: Minage réussi

Maintenant, si nous revenons à notre / endpoint blockchain et le rafraîchir, nous verrons que les deux transactions en attente ont disparu et qu'ils ont été ajoutés au bloc 4 :


```
{
  "index": 4,
  "timestamp": 1587424526972,
  "transactions": [
    {
      "amount": {
        "amount": 12.5,
        "sender": "00",
        "recipient": "30f3a2b0835a11ea8d9b6d95aec12cd5"
      }
    },
    {
      "amount": 1405,
      "sender": "NADJET-BOUCHENAFA",
      "recipient": "RIHAM-BOUDGHENE-STAMBOULI"
    },
    {
      "amount": 1405,
      "sender": "kawter",
      "recipient": "najet"
    }
  ],
  "nonce": 25275,
  "hash": "00003c6c1f0b8e382aad7f94f6d83ab7e6f2db9428682e714b671dec21ea7095",
  "previousBlockHash": "0000642946e5bc28bef7e55c1a6859590f663fcbb5e1ab40bd76e6b461ed783b"
}
```


Figure IV- 109: Fin de minage

Comme nous pouvons le voir, le bloc 4 a les trois transactions, et on a une seule transaction en attente. Cela a bien fonctionné. Si on mine un autre bloc la transaction en attente disparu et qu'il a été ajouté au bloc 5.

IV.8.6.3 Création d'un réseau décentralisé

La façon dont notre blockchain fonctionne en ce moment est que nous avons une blockchain unique, et le seul moyen d'y accéder est via l'API. Ce serveur est très centralisé, ce qui n'est pas avantageux car l'API est en contrôle total de la blockchain et des données qui s'y ajoute. Toute la technologie blockchain est hébergée sur un réseau décentralisé.

Nous allons construire un réseau de blockchain décentralisé en créant diverses instances de l'API. Chacune de ces instances de l'API va être un nœud de réseau dans notre réseau de blockchain. Tous ces nœuds fonctionneront ensemble pour héberger notre blockchain. Dans cette partie de programmation, nous couvrirons les points suivants :

- Apprendre à créer et tester plusieurs nœuds.
- Ajout de currentNodeUrl à notre réseau.
- Ajout des nouveaux nœuds finaux pour le réseau décentralisé.
- Construction / node-and-broadcast-node endpoint.
- Création et test / register-nodeendpoint.
- Ajout et test du / register-nodes-bulk endpoint.

Donc, commençons par créer notre réseau décentralisé.

IV.8.6.3.1 Création de plusieurs nœuds et l'ajout du currentNodeUrl

La première chose que nous devrons faire pour créer notre réseau décentralisé est d'apporter des modifications à notre fichier api.js.

Etape01 : Pour configurer le réseau décentralisé, nous devrons exécuter le fichier api.js plusieurs fois. Chaque fois que nous exécutons le fichier, nous voulons qu'il agisse comme un nœud de réseau. Faisons-le en exécutant le fichier sur différents ports chaque fois que nous exécutons. Nous devons faire un port variable. Pour ce faire, nous ajoutons la ligne suivante au début du code dans notre dev / api.js :

```
#const port = process.argv[2];
```

Etape02 : Accédez au fichier package.json et apportez des modifications à la commande de démarrage. Ce que nous allons faire ici, c'est aller à la fin de notre commande et passer une variable pour le numéro de port sur lequel nous voulons exécuter le nœud de réseau. Dans notre exemple, nous voulons exécuter notre

```
#"start": "nodemon --watch dev -e js dev/networkNode.js 3001"
```

Nœud de réseau pour qu'il s'exécute sur le port numéro 3001. Par conséquent, passez 3001 comme variable à la fin de la commande de démarrage :

Quelle est la variable process.argv ? Cette variable fait simplement référence à la commande de démarrage que nous exécutons pour démarrer notre serveur. La première et

le deuxième élément de la commande sont constitués de « nodemon --watch dev -e js dev/networkNode.js », et le troisième élément de la commande est la variable « 3001 ».

Etape03 : Nous voulons utiliser la variable de port. Par conséquent, dans le fichier dev / api.js, allons en bas, où nous avons mentionné le code d'écoute du port 3000 et faisons la modification suivante :

```
#app.listen(port, function() {  console.log(`Listening on port ${port}...`);
});
```

Etape04 : Exécutons le fichier api.js. Dans la fenêtre du terminal, saisissez npm start. En tapant cette commande, le serveur doit commencez à écouter le port 3001, comme nous pouvons l'observer dans la capture d'écran suivante :

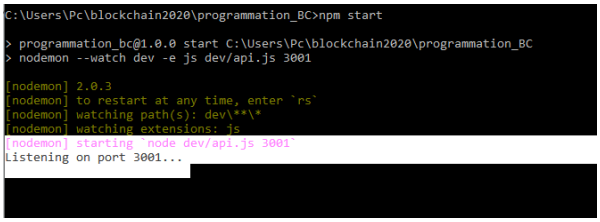


Figure IV- 110: Exécution de npm start

IV.8.6.3.2 Exécution de multiples nœuds d'api.js

La prochaine chose que nous voudrons faire est d'exécuter plusieurs nœuds de api.js. Pour cela, nous allons ajouter quelques commandes supplémentaires au fichier package.json.

Etape01 : Pour commencer, dans le fichier package.json, nous devons changer la commande "start" en "node_1". Maintenant, lorsque nous exécutons cette commande, il va démarrer notre premier nœud, qui est sur le port 3001.

Etape02 : Au lieu de taper npm start, tapez npm run node_1. Avec l'aide de cette commande, exécutez notre node_1 sur le port 3001.

Etape03 : Dans notre package.json, en tant que troisième paramètre de chacune de nos commandes, nous allons ajouter l'URL du nœud. Par conséquent, notre premier nœud a L'URL http: // localhost: 3001. Il est probable que pour notre deuxième nœud, http: // localhost: 3002. De même, vous pouvez ajouter URL des nœuds restants, comme indiqué dans la capture d'écran suivante :

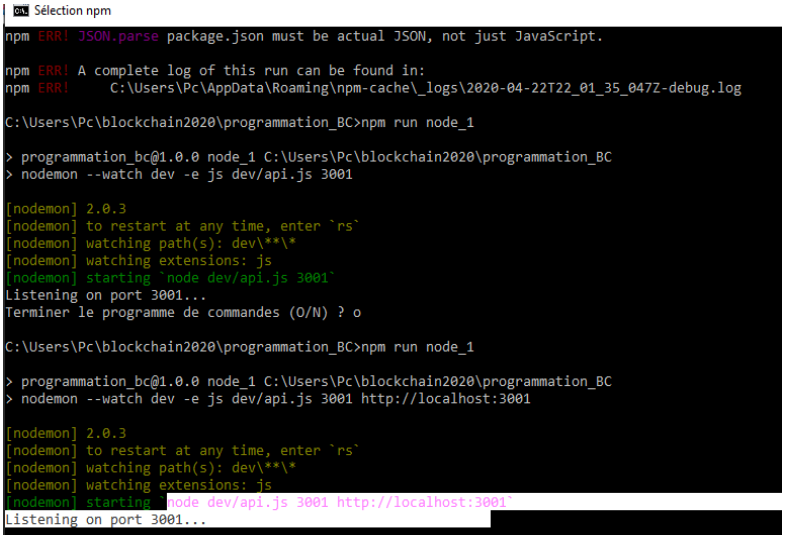


Figure IV- 111: Exécution de multiples nœuds d'api.js

Etape04 : Pour notre réseau décentralisé, nous voulons exécuter deux ou trois de ces nœuds en même temps. Pour faire ça, dupliquons la commande "node_1" : "nodemon --watch dev -e js dev / api.js 3001 http://localhost:3001" quatre fois et ensuite modifions ces commandes comme montré dans la capture d'écran suivante :

Figure IV- 112: Modifications dans le fichier package.json

Nous avons maintenant un nœud de réseau en cours d'exécution sur le port 3001 et l'autre nœud de réseau en cours d'exécution sur le port 3002. Suivez un processus similaire pour exécuter les nœuds de réseau restants sur les ports restants.

IV.9.6.3.3 Test des multiples nœuds

Maintenant, nous pourrions avoir tous les cinq nœuds du réseau en cours d'exécution. Ce que nous avons actuellement cinq nœuds de notre API en cours d'exécution, mais elles ne sont pas connectées entre eux. Pour vérifier que ces nœuds de réseau ne sont pas connectés, nous pouvons effectuer quelques tests :

Etape01 : La première transaction que nous voulons faire va être à notre nœud de réseau, qui est hébergé sur le port 3001. Alors, allons-y dans le corps et tapons les données de transaction, comme indiqué dans la capture d'écran suivante :

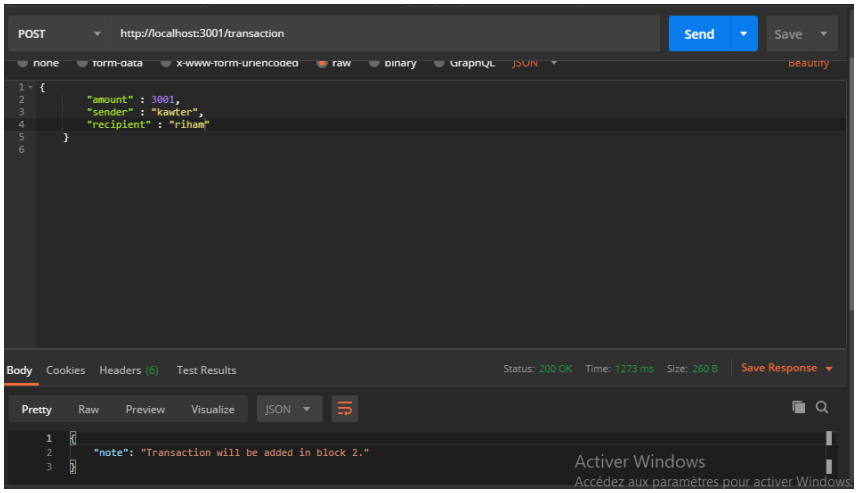


Figure IV- 113: Configuration des données de la transaction

Maintenant que nous avons envoyé la transaction donnée aux nœuds du réseau, vérifions-le. Allez dans le navigateur et allez sur localhost : 3001 / blockchain, puis appuyez sur Entrée. On aura à observer un similaire réponse, comme indiqué dans la suite capture d'écran :


```
{
  "chain": [
    {
      "index": 1,
      "timestamp": 1587594418199,
      "transactions": [],
      "nonce": 1405,
      "hash": "0",
      "previousBlockHash": "0"
    }
  ],
  "newTransactions": [],
  "pendingTransactions": [
    {
      "amount": 3001,
      "sender": "kawter",
      "recipient": "riham"
    }
  ]
}
```


Figure IV- 114: Récupération de la chaine de bloc

De la capture d'écran précédente, nous pouvons observer que nous avons une seule transaction en attente pour 3001 bitcoins. C'est l'une des transactions que nous venons d'ajoutée.

Etape02 : Maintenant, dans l'autre onglet, si nous allons à localhost : 3002 / blockchain, nous verrons que aucune transaction en attente, car nous n'avons envoyé aucune transaction à ce nœud de réseau :


```
{
  "chain": [
    {
      "index": 1,
      "timestamp": 1587594990196,
      "transactions": [],
      "nonce": 1405,
      "hash": "0",
      "previousBlockHash": "0"
    }
  ],
  "newTransactions": [],
  "pendingTransactions": []
}
```


Figure IV- 115: Création d'une transaction en attente

Etape03 : on fait les mêmes démarches de l'etape1 pour la création d'une transaction sur localhost :3003, Ensuite, si nous allons sur localhost: 3003 / blockchain, nous verrons que nous avons une transaction ici pour un montant de 3003 bitcoins

Si nous devions aller à localhost : 3004 / blockchain et localhost : 3005 / blockchain, il ne devrait pas y avoir les transactions là-bas que nous n'avons envoyé aucune transaction vers ces nœuds de réseau.

IV.8.6.3.4 L'ajoute d'un currentNodeUrl

Passons maintenant au fichier blockchain.js, et à la partie où nous définissons le const, nous allons taper ce qui suit :

```
#constcurrentNodeUrl = process.argv[3];
```

Avec cette commande, nous devrions avoir accès à l'URL du nœud actuel en utilisant la variable currentNodeUrl.

Maintenant, nous devons assigner le currentNodeUrl à notre structure de données de la blockchain. Nous le faisons en tapant la ligne de code suivante à l'intérieur de notre fonction Blockchain {} :

```
#this.networkNodes = [];
# this.currentNodeUrl = currentNodeUrl;
```

Dans les autres sections, nous remplirons ce tableau avec les URL des nœuds de tous les autres nœuds de notre réseau afin que chaque nœud soit au courant de tous les autres nœuds à l'intérieur de notre réseau blockchain.

IV.8.6.3.5 Aperçu des nouveaux endpoints

IV.8.6.3.5.1 Création de / register-and broadcast-nodeendpoint

Le / registre-node sera le prochain point de terminaison que nous ajouterons à notre réseau. Ceci est défini comme suit :

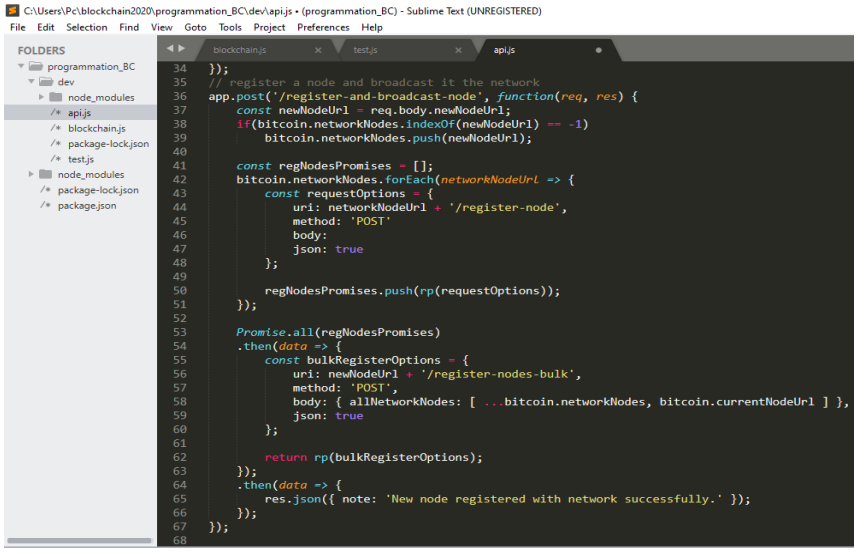


Figure IV- 116: Configuration du register node

Nous allons faire cette demande en important une nouvelle bibliothèque. Allons-y vers le terminal pour importer la bibliothèque. Dans le terminal, tapons la commande suivante :

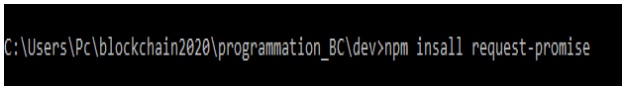


Figure IV- 117: Importation de la bibliothèque request-promise

Jusqu'à présent, nous avons enregistré le nouveau nœud avec le nœud de réseau actuel sur lequel nous sommes et nous avons diffusé le nouveau nœud à tous les autres nœuds de notre réseau. La dernière étape que nous avons effectuée à l'intérieur de ce point de terminaison est de renvoyer une réponse à celui qui l'a appelé.

- ***Explication comment le registre et nœud de diffusion fonctions :***

Nous voulons enregistrer un nouveau nœud avec notre réseau, le nœud final d'enregistrement et de diffusion est le premier point que nous voulons atteindre. La première chose qui nous faisons à l'intérieur de ce point de terminaison prend le newNodeUrl et l'enregistre avec le nœud actuel en le poussant dans notre tableau networkNodes.

La prochaine étape que nous devons faire est de diffuser ce newNodeUrl au reste des nœuds de notre réseau. Nous le faisons à l'intérieur de la boucle forEach. Tout ce qui se passe à l'intérieur de cette boucle est que nous faisons une demande à chacun des autres nœuds de notre réseau. Nous faisons cette demande au point d'extrémité de nœud de registre. Une fois toutes ces demandes terminées sans aucune erreur, nous pouvons supposer que le newNodeUrl a été enregistré avec succès avec tous nos autres nœuds de réseau.

IV.8.6.3.5.2 Construire et tester le / register-nodeendpoint

Maintenant que nous avons construit le / register-and -broadcast-node final, il est temps de passer à certaines choses qui sont un peu moins complexes. Dans cette section, commençons la construction du register-nodeendpoint. Ça va être très simple par rapport au point final que nous avons construit dans la partie précédente.

Ce nœud final registre-node est l'endroit où chaque nœud dans le réseau va recevoir la diffusion qui est envoyée par le endpointregister-and-broadcast-node ; ce point registre-node doit faire s'inscrire le nouveau nœud avec le nœud qui reçoit la demande. Pour commencer à créer le endpointregister-node, procédons comme suit :

```
69
70  // register a node with the network
71  app.post('/register-node', function(req, res) {
72      const newNodeUrl = req.body.newNodeUrl;
73      const nodeNotAlreadyPresent = bitcoin.networkNodes.indexOf(newNodeUrl) == -1;
74      const notCurrentNode = bitcoin.currentNodeUrl !== newNodeUrl;
75      if (nodeNotAlreadyPresent && notCurrentNode) bitcoin.networkNodes.push(newNodeUrl);
76      res.json({ note: 'New node registered successfully.' });
77  });
78
```

Figure IV- 118: Construit le / register-and -broadcast-node

- *Test de register-node*

Testons-le / register-nodeendpoint pour s'assurer qu'il fonctionne correctement et pour mieux comprendre son fonctionnement.

- *Installation de la bibliothèque request*

Maintenant, pour tester le endpoint que nous venons de créer, il est nécessaire d'installer la bibliothèque request selon la version de la bibliothèque de request-promise disponible. Pour installer la bibliothèque de requêtes, allons simplement sur notre terminal, et à l'intérieur nous exécutez la commande suivante :

```
C:\Users\Pc\blockchain2020\programmation_BC\dev>npm install request --save
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN saveError ENOENT: no such file or directory, open 'C:\Users\Pc\blockchain2020\programmation_BC\dev\package.json'
npm WARN enoent ENOENT: no such file or directory, open 'C:\Users\Pc\blockchain2020\programmation_BC\dev\package.json'
npm WARN dev No description
npm WARN dev No repository field.
npm WARN dev No README data
npm WARN dev No license field.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.1.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.1.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ request@2.88.2
added 42 packages from 51 contributors and audited 1038 packages in 157.249s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Figure IV- 119: Installation de la bibliothèque request

- *Test*

Pour commencer, nous allons taper http: // localhost: 3001 / register-node dans la barre d'adresse, comme illustre la capture d'écran ci-dessous :

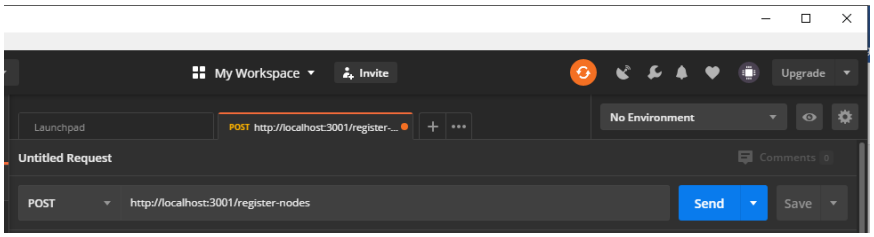


Figure IV- 120: Test de register-node

Ensuite, à l'intérieur de la zone de texte, créons un objet et ajoutons le code suivant :

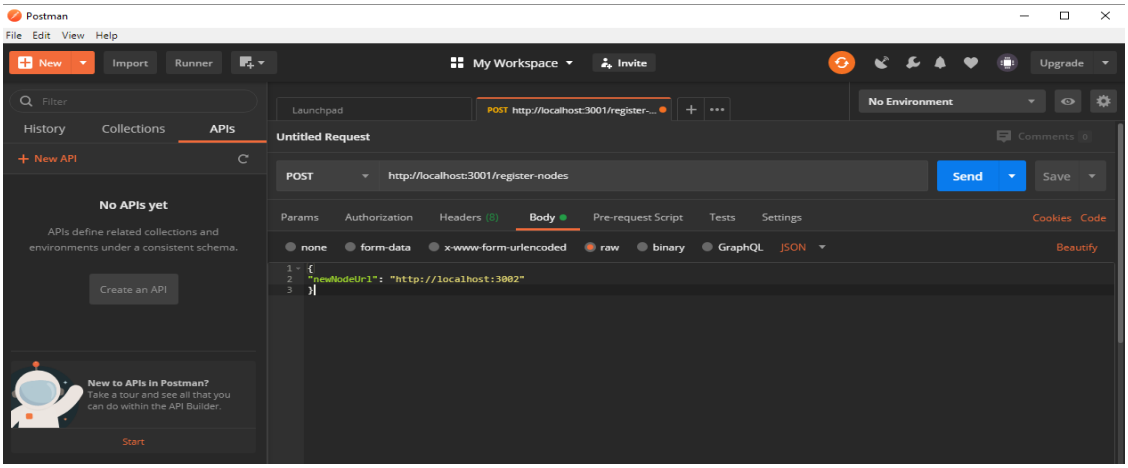


Figure IV- 121: Ajout de node url

Pour vérifier cela, allons à Postman et cliquons sur le bouton Envoyer. Nous obtiendrons la réponse Nouveau nœud enregistré avec succès. Maintenant, passons à notre navigateur et tapons localho st: 3001 / blockchain dans la barre d'adresse, puis appuyons nous sur Entrée. Nous verrons une sortie similaire à ce qui est montré dans la capture d'écran suivante :

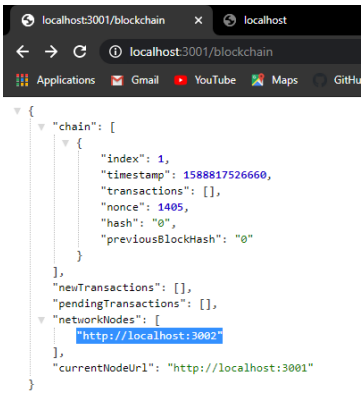


Figure IV- 122: Obtention de la réponse Nouveau nœud enregistré avec succès

Puisque nous venions d'enregistrer notre deuxième nœud avec notre nœud actuel sur localhost: 3001, nous avons l'URL de notre deuxième nœud à l'intérieur de cet tableau.

IV.8.6.3.5.3 Construire le / register-nodes-bulk endpoint

Le prochain endpoint que nous allons construire s'appelle register-nodes-bulk. Ces trois points sur lesquels nous avons travaillé pour créer notre réseau de blockchain décentralisé.

Le register-nodes-bulk accepte des données contenant les URL de chaque nœud déjà présent dans le réseau. Ensuite, nous allons simplement enregistrer tous ces nœuds de réseau avec le nouveau nœud. Le nouveau nœud est le nœud sur lequel le register-nodes-bulk est atteint.

Pour créer le endpointregister-nodes-bulk, nous devrons faire l'hypothèse que toutes les URL de nœuds qui sont actuellement dans notre réseau sont transmises en tant que données et que nous pouvons les accéder via la propriété req.body.allNetworkNodes. C'est parce que nous envoyons les données allNetworkNodes lorsque nous appelons cela

endpoint dans le bloc Promise.all (regNodesPromise). Ici, nous envoyons allNetworkNodes au endpointregister-nodes-bulk. Cela nous donnera accès aux données allNetworkNodes à l'intérieur du endpoint. Ajoutons la ligne de code suivante à notre endpointregister-nodes-bulk.

```
78  // register multiple nodes at once
79  app.post('/register-nodes-bulk', function(req, res) {
80      const allNetworkNodes = req.body.allNetworkNodes;
81      allNetworkNodes.forEach(networkNodeUrl => {
82          const nodeNotAlreadyPresent = bitcoin.networkNodes.indexOf(networkNodeUrl) == -1;
83          const notCurrentNode = bitcoin.currentNodeUrl !== networkNodeUrl;
84          if (nodeNotAlreadyPresent && notCurrentNode) bitcoin.networkNodes.push(networkNodeUrl);
85      });
86
87      res.json({ note: 'Bulk registration successful.' });
88  });_
89
```

Figure IV- 123: Construire le / register-nodes-bulk endpoint

Fondamentalement, tout ce que nous déclarons dans la déclaration if est que pendant que nous parcourons chaque nœud de réseau que nous ajoutons, si ce nœud n'est pas déjà présent dans notre réseau de nœuds de réseau et si cela le nœud n'est pas l'URL de notre nœud actuel, alors nous voulons ajouter le networkNodeUrl à notre tableau networkNodes. Une fois que nous avons terminé la boucle forEach, nous aurons enregistré tous les nœuds du réseau qui sont déjà présents à l'intérieur de notre réseau blockchain. Tout ce que nous avons à faire à ce stade est de renvoyer une réponse, comme suit.

- *Test des / register-nodes-bulk :*

Nous allons tester notre register-nodes-bulk endpoint pour nous assurer que cela fonctionne correctement. Cela nous permettra de comprendre comment cela fonctionne :

Etape01 : Nous allons diriger vers Postman. Ici, nous allons tapper le localhost: 3001 / register-nodes-bulk. Lorsque nous testons ce endpoint, nous nous attendons à recevoir des données, qui est le tableau allNetworkNodes.

Etape02 : dans l'onglet corps de Postman, avec l'option raw et JSON (application / json) format sélectionné pour le texte, nous ajoutons les lignes de code suivantes au corps :

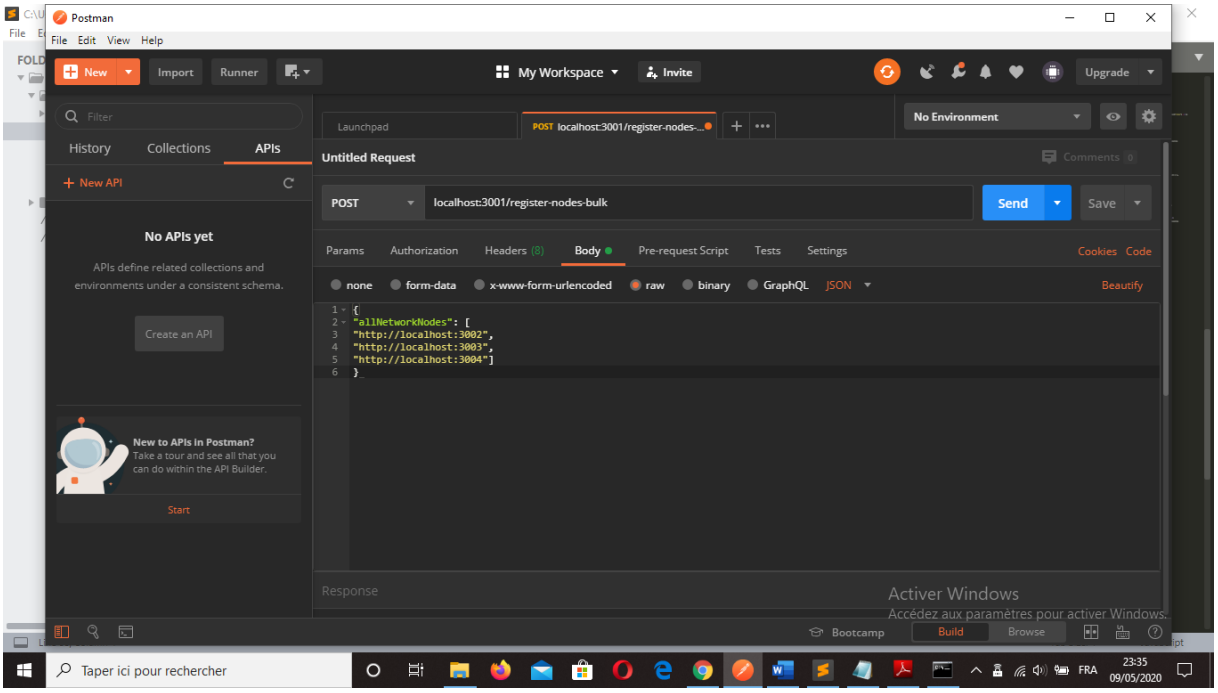


Figure IV- 124: Test des / register-nodes-bulk

Lorsque nous exécutons cette demande maintenant, nous devrons enregistrer ces trois URL avec notre nœud qui s'exécute localhost: 3001. Voyons voir si ça marche. Cliquons sur le bouton Send et nous recevrons une réponse qui indique l'enregistrement en bulk réussi comme le montre la capture d'écran suivante :

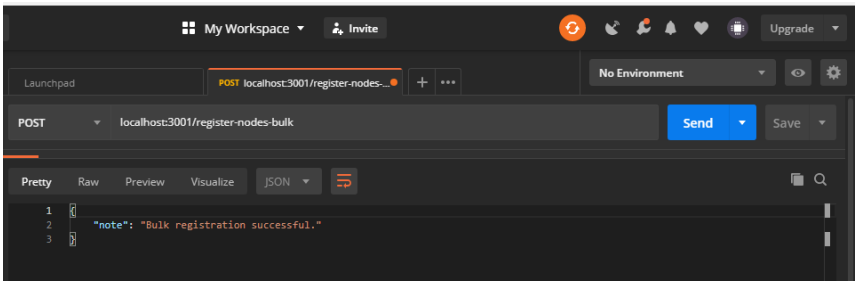


Figure IV- 125: Recevoir une réponse qui indique l'enregistrement en bulk réussi

Maintenant, si nous nous dirigeons vers le navigateur, nous pouvons vérifier que cela a fonctionné. Dans la barre d'adresse, tapons localhost: 3001 / blockchain puis appuyons sur Entrée. Nous arriverons à observer les trois URL qui ont été ajoutées à l'intérieur du tableau networkNodes, car ce sont enregistrées en bulk :

Figure IV- 126: Ajout réussie de trois url

Donc, il ressemble que notre endpointregister-node-bulk fonctionne comme il se doit.

IV.9 Conclusion

Dans ce chapitre, nous avons commencé par construire un réseau bancaire avec un routage amélioré, sécurisé et rendant le trafic plus efficace. Pour répondre à ce besoin de communication sécurisé, nous avons utilisé un réseau privé virtuel (VPN) avec ses différentes topologies ainsi que les détails sur le protocole IPsec, sans oublier le GRE tunnel qui est un des mécanismes de tunneling. Le trafic qui traverse ce tunnel doit être protégé lors de l'échange de la monnaie. Également dans ce chapitre, nous avons également appris à construire la fonction constructeur, puis nous sommes passés à la création de méthodes étonnantes telles que createNewBlock, creatNewTransaction, getLastBlock, etc. Nous avons ensuite découvert la méthode de hachage, le hachage SHA256, et créé une méthode pour générer un hachage pour nos données de bloc. Nous avons également appris ce qu'est une preuve de travail et comment cela fonctionne. Dans ce chapitre, nous avons également appris à tester les différentes méthodes que nous avons créées et vérifier si elles fonctionnent comme prévu. Dans la deuxième partie ; nous avons appris à configurer une blockchain. Puis, nous avons appris à configurer une blockchain Express.js dans notre projet, ainsi que comment utilisez-le pour construire notre API / serveur. Ensuite nous installé Postman et compris comment utilisez-le pour tester nos endpoints. Après cela, nous passé à construire divers points d'extrémité de notre serveur et testé ceux pour vérifier si ou non, ils fonctionnaient correctement. Dans la prochaine partie, nous allons créer un réseau de nœuds ou réseau décentralisé pour héberger notre blockchain, tout comme ceux qui sont hébergés dans le monde réel. Jusqu'à présent, vous avez accompli beaucoup de choses dans ce Projet Fin d'étude. Nous avons créé une blockchain décentralisée réseau qui fonctionne actuellement sur cinq nœuds, et nous avons créé la fonctionnalité pour synchroniser l'ensemble du réseau, afin que tous les nœuds ont exactement les mêmes données.

Conclusion générale

Avec la façon dont la technologie blockchain prend rapidement le contrôle d'un certain nombre de nos secteurs existants en tant que moyen de transaction actif et plus préférable, il n'est pas difficile d'imaginer un avenir lorsque la technologie blockchain régnerait sur tous les autres moyens de contrat. En fait, l'avenir de blockchain est extrêmement brillant non seulement dans le secteur bancaire et financier, mais dans tous les autres secteurs imaginables dans la société.

La technologie de la blockchain est en constante évolution ; chaque fois qu'un nouveau besoin apparaît, ce concept change pour répondre à ce besoin. Ceci est encore un autre signe que blockchain a la capacité de façonner ses idéaux pour répondre aux besoins de l'avenir, et a une brillante opportunité de le faire dans les prochaines années. La technologie de la blockchain est parfois, et à juste titre, appelée la « seconde venue d'Internet ». Mais pour le profane, la blockchain pourrait ne ressembler qu'à un grand livre géant, qui prend en compte chaque transaction effectuée dans le monde entier. Mais comme vous pouvez le voir maintenant, cela peut être bien plus.

Dans le secteur bancaire, bien sûr, la technologie blockchain est sur le point d'apporter d'énormes changements très bientôt. Les experts estiment qu'avant longtemps, de nombreuses grandes banques et institutions financières commenceront à adopter la Blockchain au lieu d'utiliser la monnaie traditionnelle. Lorsque cela se produira, les systèmes bancaires mondiaux changeront pour toujours ; les cyber-risques en cas de transferts d'argent internationaux seront réduits, et chaque transaction deviendra plus rapide et sans effort.

L'étude et la conception d'un réseau bancaire sécurisé par la technologie blockchain a fait l'objet de notre travail de recherche dans le cadre de ce projet de fin d'études. Pour cela, nous nous sommes intéressés, aux fondements même de cette technologie, en analysant les principes essentiels qui la régissent. Ensuite, nous avons fait un zoom sur l'industrie bancaire, pour mieux constater comment cette technologie promettait de remettre en cause le mode de fonctionnement des banques et nous avons vu comment celles-ci se sont unies pour contrecarrer cette menace.

Cette technologie a permis aux banques de se rapprocher via la formation de consortiums. Les objectifs de ces associations étaient multiples. Il s'agissait en premier lieu de créer des synergies communes afin de faire face à l'arrivée de nouveaux acteurs qui promettaient de mettre à mal le modèle bancaire en supprimant de nombreux intermédiaires. Dans un second temps l'objectif des consortiums était de tirer profit de ces applications en les déployant au sein de leurs structures et nous avons vu que de nombreux établissements ont d'ores et déjà pris de l'avance sur le sujet.

Aussi, nous avons parlé des formidables opportunités que représente la blockchain pour les banques à savoir : des économies de temps, économies de coût, réductions des risques réglementaires et enfin, une plus grande traçabilité des opérations laissant notamment espérer à terme la mort du blanchiment d'argent.

Ainsi, les perspectives futures sont dans un premier temps l'installation du GNS3 parce que d'habitude si on veut configurer un équipement réseaux on utilise command line Soit putty ou bien secure shell, mais maintenant on peut aussi faire la configuration en utilisant Python Script au lieu de lancer cli on crée notre propre Python script, et dans seconds temps programmer la blockchain avec langage python pour vous pouvez voir réellement l'application.

Bibliographie

[1]*https://www.youtube.com/watch?v=ZbHLNinXy9E&feature=youtu.be*date #10/01/2020

[2]*https://www.hbrfrance.fr/chroniques-experts/2018/05/20131-breve-histoire-de-blockchain/#* #10/01/2020

[3] GANNE, Emmanuelle. *Can Blockchainrevolutionize international trade?*.Geneva : World Trade Organization, 2018.Date 10/01/2020

[4]https://www.memoireonline.com/10/17/10113/L-impact-des-technologies-de-l-information-et-de-la-communication-sur-l-entreprise.html #10/02/2020

[5]https://www.pwc.ru/ru/publications/blockchain.html #10/02/2020

[6] https://blockchainfrance.net/decouvrir-la-blockchain/c-est-quoi-la-blockchain/ #10/02/2020

[7] Crosby, Micheal; Nachiappan; Pattanayak, Pradhan; Verma, Sanjeev; Kalyanaraman, Vignesh, "Blockchain Technology: Beyond Bitcoin," Berkeley, CA: SutardjaCenter for Entrepreneurship & Technology, University of California, http://scet.berkeley.edu/wp-content/uploads/BlokchinPaper.pdf , October 16.2015

[8]http://m.qukuaiwang.com.cn/news/4816.html#12/02/2020

[9]https://www.blocktempo.com/understand-structure-of-blockchain-bitcoin/#12/02/2020

[10]https://blockgeeks.com/guides/fr/contrats-intelligents #12/02/2020

[11] https://blog.toright.com/posts/5981/pow-pos-dpos-consensus-intro.html #12/02/2020

[12]https://www.chainnews.com/articles/604126127022.htm #12/02/2020
[13] https://moussakayre.digital/quest-ce-que-la-technologie-blockchain-un-guide-pour-debutants/
[14]https://www.journaldunet.com/economie/finance/1195520-blockchain-avril-2019/
[15]http://www.wikiwai.com/2020/02/12/comprehension-et-conception-de-services-avec-la-blockchain-bitcoin-ethereum/ #13/02/2020
[16]http://www.senat.fr/rap/r17-584/r17-584_mono.html #13/02/2020
[17] http://dspace.univ-tlemcen.dz/bitstream/112/9319/1/Partage-de-fichiers-de-pair-a-pair-sur-JXTA.PDF#13/02/2020
[18] https://www.slideshare.net/AmineHAMOUDA/prsentation-blockchain-v2 #13/02/2020
[19] https://www.lacircum.com/index.php/fr/la-crypto-monnaie-pour-les-nuls-debutants/6-blockchain-definition-expliquee #13/02/2020
[20] http://www.planet-libre.org/index.php?post_id=19524 #13/02/2020
[21] https://www.blockchains-expert.com/blockchain-privee-vs-blockchain-publique/ #13/02/2020
[22] https://cryptoast.fr/blockchain-chaine-de-blocs/ #13/02/2020
[23] https://solutions.lesechos.fr/tech/c/part-v-blockchain-privee-publique-difference-9229/ #13/02/2020
[24] https://www.le-vpn.com/fr/cest-quoi-le-vpn-ou-reseau-prive-virtuel/ #13/02/2020
[25] https://www.cactusvpn.com/fr/beginners-guide-to-vpn/quest-ce-quopenvpn/ #15/02/2020
[26] https://www.purevpn.fr/quest-ce-quun-vpn/protocoles/ikev2 #15/02/2020
[27] https://www.cisco.com/c/fr_ca/support/docs/security-vpn/ipsec-negotiation-ike-protocols/14106-how-vpn-works.html #15/02/2020
[28] https://www.purevpn.fr/quest-ce-quun-vpn/protocoles/sstp #15/02/2020
[29] https://www.purevpn.fr/quest-ce-quun-vpn/protocoles#15/02/20^ #17/02/2020

[30] https://www.bbc.com/zhongwen/simp/business-43089926

[31] https://www.forbes.com/forbes/2011/0509/technology-psilocybin-bitcoins-gavin-andresen-crypto-currency.html#32af59f0353e

[32] https://journals.openedition.org/regulation/11489

[33] https://www.investopedia.com/terms/a/altcoin.asp

[34]https://fr.cryptonews.com/guides/how-to-get-altcoins.plusieurs%20crypto-monnaies%20similaires%20ont%20%C3%A9t%C3%A9%20cr%C3%A9%C3%A9es,%20souvent%20appel%C3%A9es%20altcoins

[35]http://finances-et-patrimoine.fr/bourse/la-crypto-monnaie/

[36]http://www.xinhuanet.com//fortune/2017-06/29/c_129643337.htm

[37] https://crypto-monnaie.pro/quest-ce-que-la-crypto-monnaie/

[38] https://www.capital.fr/entreprises-marches/bitcoin-ethereum-ripple-lavis-de-capital-sur-les-cryptomonnaies-qui-comptent-1278913

[39] https://coin24.fr/crypto-monnaies/differences-btc-eth/

[40] https://conseilscrypto.com/cest-quoi-litecoin-ltc/

[41] https://www.phonandroid.com/bitcoin-vs-ethereum-difference.html

[42] https://coin24.fr/crypto-monnaies/differences-btc-eth/

[43] https://coin24.fr/crypto-monnaies/

[44] LANDAU, J. P. et GENAIS, A. Les crypto-monnaies, rapport au Ministre de l'Économie et des Finances, 4 July. 2018.

[45] https://www.journaldunet.fr/patrimoine/guide-des-finances-personnelles/1207712-cryptomonnaie/ #18/03/20 13h17

[46] https://www.futura-sciences.com/tech/definitions/informatique-cryptomonnaie-18278/ #18/03/20 13h23

[47] https://www.boj.or.jp/announcements/education/oshiete/money/c27.htm/ #18/03/20 14h01

[48]https://www.supinfo.com/articles/single/10012-cryptomonnaie-fonctionnement-definitions #18/03/20 14h15

[49] https://www.cafedelabourse.com/archive/article/bitcoins-monnaie-virtuelle-investir-crypto-monnaie

[50] http://www.synergix.ch/fr/info/news/fintech-lexique-comprendre-univers-blockchain-et-crypto-monnaies

[51] https://www.investopedia.com/terms/b/blockchain.asp

[52] https://theinnovationandstrategyblog.com/2018/05/22/levolution-du-contexte-reglementaire-bancaire-amene-les-banques-accelerer-leur-digitalisation-2-2/

[53] LANDAU, J. P. et GENAIS, A. Les crypto-monnaies, rapport au Ministre de l'Économie et des Finances, 4 July. 2018.

[54] https://fr.wikipedia.org/wiki/Cryptographie #19/03/2020-19h10

[55] https://www.clicours.com/cours-cryptographie-les-fondements-de-la-cryptographie/ #19/03/2020-19h40

[56] https://lipn.univ-paris13.fr/~poinsot/SEC/Chapitre_7_Printable.pdf #19/03/2020-21h50

[57] https://www.binance.vision/fr/security/what-is-symmetric-key-cryptography #21/03/20 23h02

[58] https://waytolearnx.com/2018/07/difference-entre-le-chiffrement-par-bloc-et-le-chiffrement-par-flot.html #21/03/20 23h07

[59] https://mathworld.wolfram.com/XOR.html #21/03/20 23h41

[60] https://www.geeksforgeeks.org/data-encryption-standard-des-set-1/ #21/03/20 23h45

[61]https://www.comparitech.com/blog/information-security/what-is-aes-encryption/ #21/03/2020-23h56

[62] https://waytolearnx.com/2018/07/difference-entre-des-et-aes.html 22/03/20 00h00 #22/03/2020-00h30

[63]https://www.frenchweb.fr/petite-histoire-de-la-cryptographie-sur-la-piste-de-la-cryptographie-asymetrique/270507 #22/03/0220-1h20

[64] https://www.globalsign.fr/fr/centre-information-ssl/cryptographie-cle-publique/#22/03/2020-23h57

[65]https://www.di.ens.fr/~nitulesc/files/crypto3.pdf#22/03/0220-12h10

[66]https://fr.wikipedia.org/wiki/Cryptographie_asym%C3%A9trique#22/03/0220-14h15

[67] Singhal, B., Dhameja, G., & Panda, P. S. (2018). *Beginning Blockchain: A Beginner's Guide to Building Blockchain Solutions*. Apress

[68] https://www.silicon.fr/hub/malwarebytes-hub/la-signature-electronique-garant-de-lintegrite-de-vos-donnees #22/03/0220-15h20

[69] https://segmentfault.com/a/1190000012158045 #22/03/0220-15h30

[70] https://fr.wikipedia.org/wiki/Elliptic_curve_digital_signature_algorithm #22/03/0220-17h10

[71] https://waytolearnx.com/2018/07/difference-entre-le-cryptage-symetrique-et-asymetrique.html #22/03/20 20h27

[72] https://baike.baidu.com/item/hash/390310 #22/03/20 22h47

[73] https://fr.wikipedia.org/wiki/Fonction_de_hachage_cryptographique #22/03/20 22h54

[74] https://www.md5hashgenerator.com/ #22/03/20 22h57

[75] https://www.geeksforgeeks.org/sha-1-hash-in-java/ #22/03/20 23h07

[76] https://www.geeksforgeeks.org/difference-between-md5-and-sha1/ #22/03/20 23h11

[77] https://fr.wikipedia.org/wiki/Longueur_de_cl%C3%A9 #22/03/20 23h13

[78] https://happypeter.github.io/binfo/all-pub #22/03/20 23h23

[79] https://codertw.com/%E5%89%8D%E7%AB%AF%E9%96%8B%E7%99%BC/246717/ #22/03/20 23h25

[80] https://hal.archives-ouvertes.fr/hal-01778949/document

[81] https://www.binance.vision/fr/blockchain/what-are-nodes

[82] https://www.esds.co.in/blog/everything-need-know-blockchain/#sthash.22bsmU8y.en54I8Ke.dpbs

[83] https://klmoney.wordpress.com/bitcoin-dissecting-transactions-part-2-building-a-transaction-by-hand/

[84] https://www.fullstackweekly.com/category/bitcoin/how-does-bitcoin-work-part-3-the-billcoin-transactions

[85] https://cryptoast.fr/bloc-blockchain-crypto-explication/

[86] https://fr.wikipedia.org/wiki/Minage_de_cryptomonnaie

[87] http://blog.hubwiz.com/2019/01/03/blockchain-7step-miner/

[88]https://siecledigital.fr/2016/10/17/blockchain-mais-qui-sont-les-mineurs/

[89]https://www.universalis.fr/encyclopedie/reseaux-informatiques/6-securite-dans-les-reseaux/ #16/03/2020-23h23

[90]http://big5.baike.qianzhan.com/detail/bk_2286e38b.htmll #16/02/2020-23h34

[91]https://www.pedagogie.ac-aix-marseille.fr/upload/docs/application/pdf/2012-07/formation_reseau.pdf #17/03/2020-00h16

[92]https://www.ionos.fr/digitalguide/serveur/know-how/les-types-de-reseaux-informatiques-a-connaitre/ #17/03/2020-11h57

[93]https://fr.qwe.wiki/wiki/Internet #17/03/2020-12h03

[94] https://fr.wikipedia.org/wiki/Intranet #17/03/2020-12h07

[95]https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203385-extranet-definition/ #17/03/2020-12h14

[96]https://www.supinfo.com/articles/single/7300-differentes-topologies-reseaux#17/03/2020-12h25

[97]http://sadalahsadalah.e-monsite.com/medias/files/chapitre5-telephonie.docx.pdf #17/03/2020-13h55

[98]https://spip.teluq.ca/inf1160/IMG/pdf/inf1160-notionsfondamentales.pdf#17/03/2020- 14h13

[99]https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203405-tcp-ip-transmission-control-protocol-internet-protocol-definition-traduction/ #17/03/2020-14h18

[100]https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203389-ip-adresse-ip-internet-protocol-definition/ #17/03/2020-14h22

[101]https://www.ionos.fr/digitalguide/serveur/know-how/quest-ce-que-linternet-protocol-definition-dip/ #17/03/2020-14h26

[102]https://baike.baidu.com/item/%E8%B7%AF%E7%94%B1%E5%8D%8F%E8%AE%AE #30/08/20-13h20

[103]https://zh.wikipedia.org/wiki/%E8%B7%AF%E7%94%B1%E5%8D%8F%E8%AE%AE #30/08/20-13h31

[104] https://blog.csdn.net/qq_30852577/article/details/79009003 #30/08/20-13h41

[105] https://www.sohu.com/a/116995737_465914 #30/08/20-14h10

Annexe

Annexe 1 Installation d'Apache

1. Qu'est-ce qu'Apache :

Apache est le logiciel de serveur Web le plus populaire. Il permet à un ordinateur d'héberger un ou plusieurs sites Web accessibles sur Internet à l'aide d'un navigateur Web. La popularité d'Apache sur le marché de l'hébergement Web est largement due au fait qu'il est open source et gratuit à utiliser.

Apache prend également en charge plusieurs plates-formes, y compris les systèmes d'exploitation Linux, Windows et Macintosh.

Apache peut héberger des sites Web statiques, ainsi que des sites Web dynamiques qui utilisent des langages de script côté serveur, tels que **PHP**, **Python** ou **Perl**. La prise en charge de ces langues et d'autres est implémentée via des modules ou des packages d'installation qui sont ajoutés à l'installation Apache standard. Apache prend également en charge d'autres modules, qui offrent des options de sécurité avancées, des outils de gestion de fichiers et d'autres fonctionnalités. La plupart des installations Apache incluent un module de réécriture d'URL appelé " **mod_rewrite**", qui est devenu un moyen courant pour les webmasters de créer des URL personnalisées.

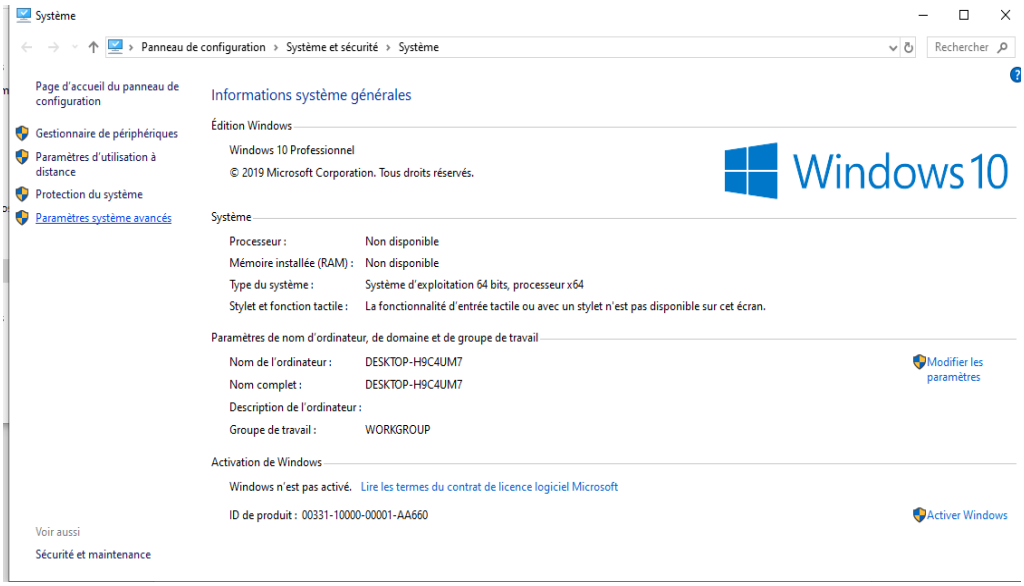
Bien que le logiciel serveur Web Apache soit communément appelé simplement « Apache », il est techniquement appelé « serveur HTTP Apache », car le logiciel sert des pages Web via le protocole **HTTP**. Quand Apache est en cours d'exécution, son nom de processus est « **httpd** », qui est l'abréviation de « **démon http** ».

Comment installer Apache :

Tout d'abord, nous devons télécharger Apache à partir du site Web actuel d'Apache « https://www.apachelounge.com/download/ »

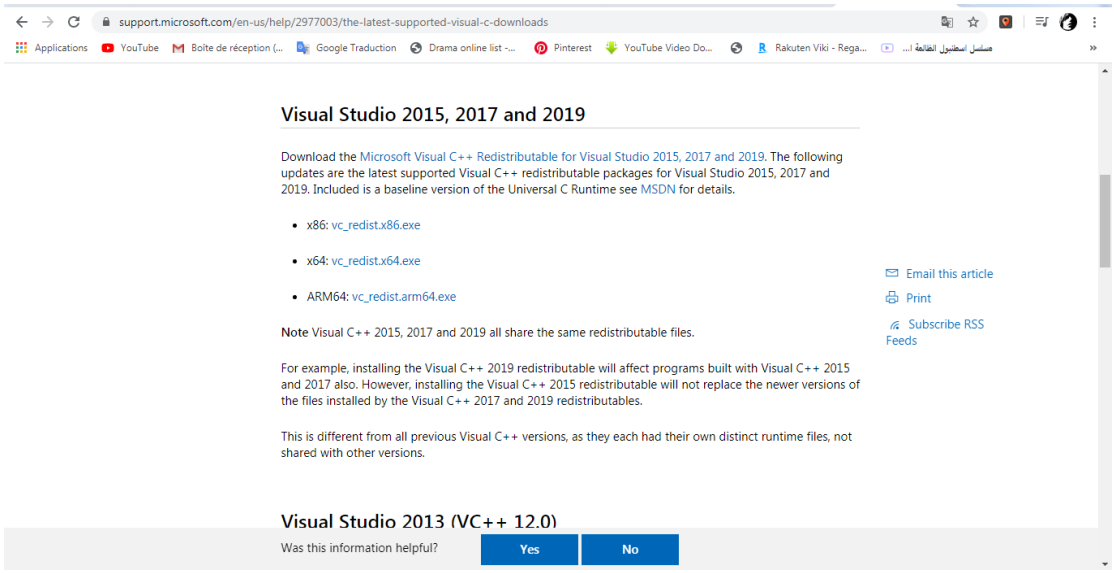
Et nous téléchargeons la dernière version d'apache qui est actuellement 2.4.43, nous pouvons télécharger le 64 bits pour la version ou le 32 bits [pour vérifier quel type desystème nous avons, nous pouvons faire un clic droit sur l'icône Windows et cliquez sur

système dans ce panneau, nous pouvons vérifier si notre type de système est un système d'exploitation en version 64 bits ou 32 bits.]



Alors maintenant nous téléchargeons le fichier zip d'Apache.

Après son téléchargement, nous pouvons accéder au dossier de téléchargement, nous pouvons trouver ici le zip, double-cliquez dessus pour l'extraire et à l'intérieur, nous trouverons un dossier apache 24, maintenant nous devons accéder à notre PC et à notre pilote C et copier / coller le dossier apache 24 à la racine de base de notre lecteur C après avoir transféré le dossier entier, nous devons télécharger un autre cadre important pour utiliser apache et ce cadre est Microsoft visuel C ++ téléchargeons la dernière version qui est actuellement 2019 (à partir de ce site https://support.microsoft.com/en-us/help/2977003/the-latest-supported-visual-c-downloads)



Faites défiler ici et nous choisissons la version de notre Windows (32bits (x86) ou 64bits), après téléchargement est en cours automatique.

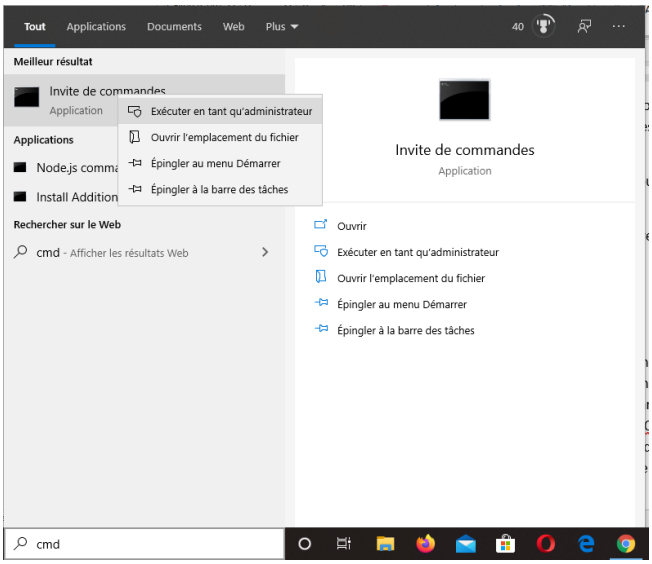
Après le téléchargement, accédez à nouveau à notre dossier de téléchargement et double-cliquez sur l'application que vous venez de télécharger puis cliquez sur **Exécuter** et le téléchargement a commencé.

Visual C++ Redistributable Package est un package Microsoft C++ components qui permet de faire tourner des applications Visual C++.

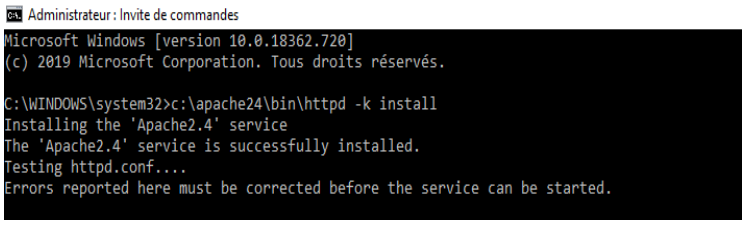
Lorsque le Visual C++ Redistributable Package est mal installé ou non présent sur l'ordinateur, vous pouvez rencontrer des messages XXXX.DLL est manquant ou Erreur api-ms-win-crt-runtime-l1-1-0.dll

En clair donc, ces packages Visual C++ sont très importants pour que les applications puissent fonctionner correctement.

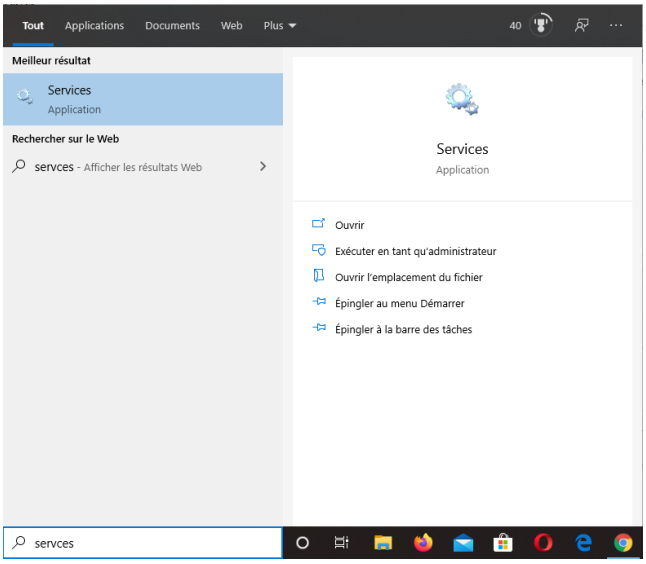
Puis on installe apache comme une invite de commande d'ouverture de service windows , on fait un clic droit sur l'aplication et exécutez en tant qu'administrateur puis on clique sur yes.



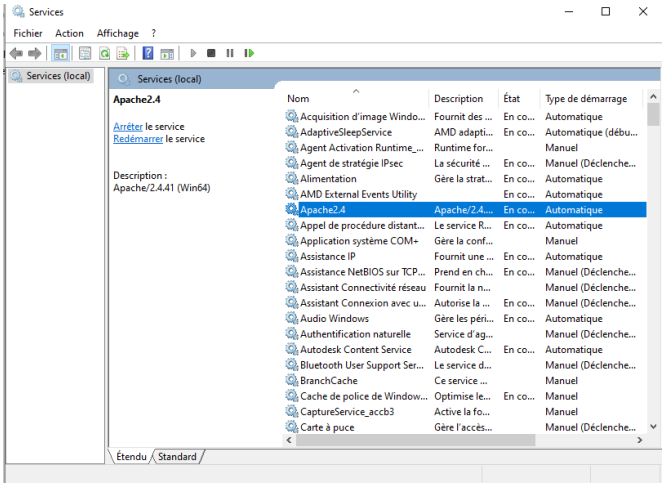
Et maintenant tappons httpd -k install



Apres nous allons vérifier le service Apache. Cliquons sur l'icon de windows et tappons services

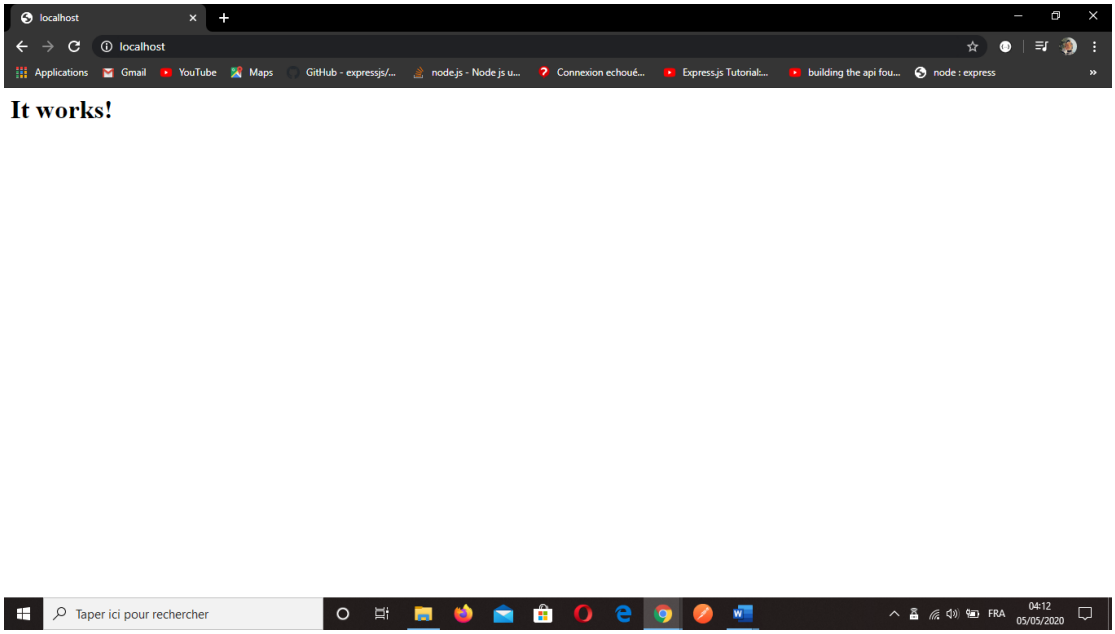


Maintenant, on ouvre l'application puis on cherche le service nomé Apache24



Nous pouvons démarrer le serveur Apache, faites un clic droit dessus et cliquer sur démarrer.

Quand je fais 127.0.0.1 (localhost) s'affiche dans la fentre It works !



Ca veut dire que mon serveur fonctionne bien.

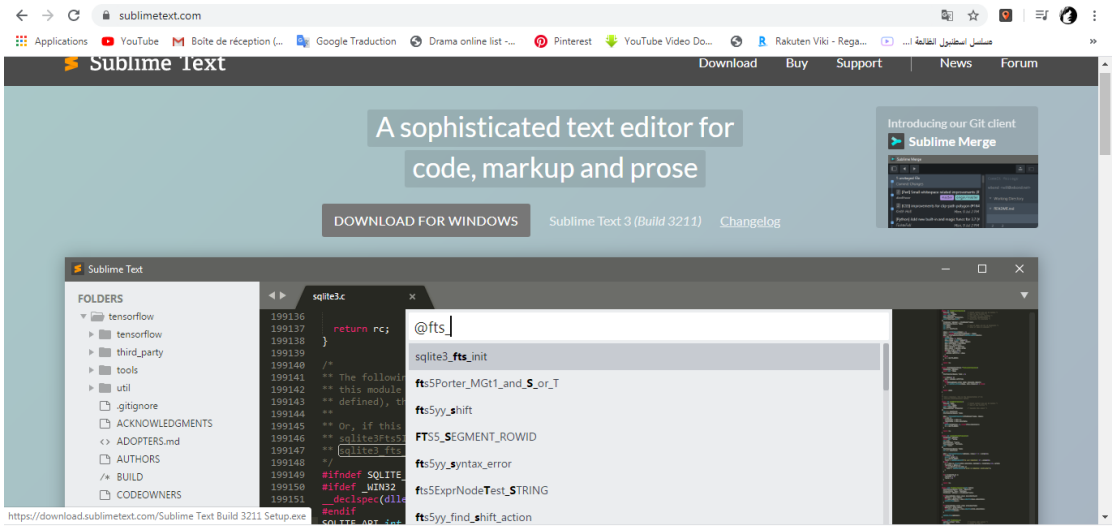
Annexe 2 Installation du Sublime

1. **Introduction :**

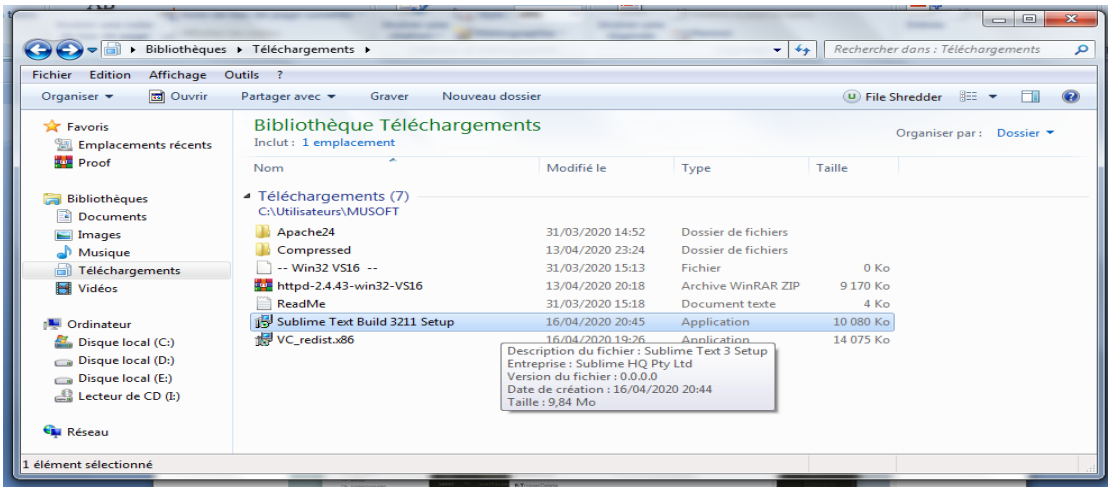
Sublime Text est un éditeur de texte vraiment puissant. Disponible à la fois sur Windows, sur Mac et sur Linux, il est conçu pour prendre en charge plusieurs langages de programmation variés allant du langage de programmation C à l'Action Script en passant par les langages PHP, Objective-C ou encore OCaml voire même du Scripting comme le Shell Scripting ou encore le SQL. C'est un éditeur à tout faire. Ce qui fait réellement sa force est donc cette capacité à prendre en charge de nombreux langages mais aussi d'apporter de nombreuses fonctionnalités pratiques qui faciliteront la création de code pour les développeurs.

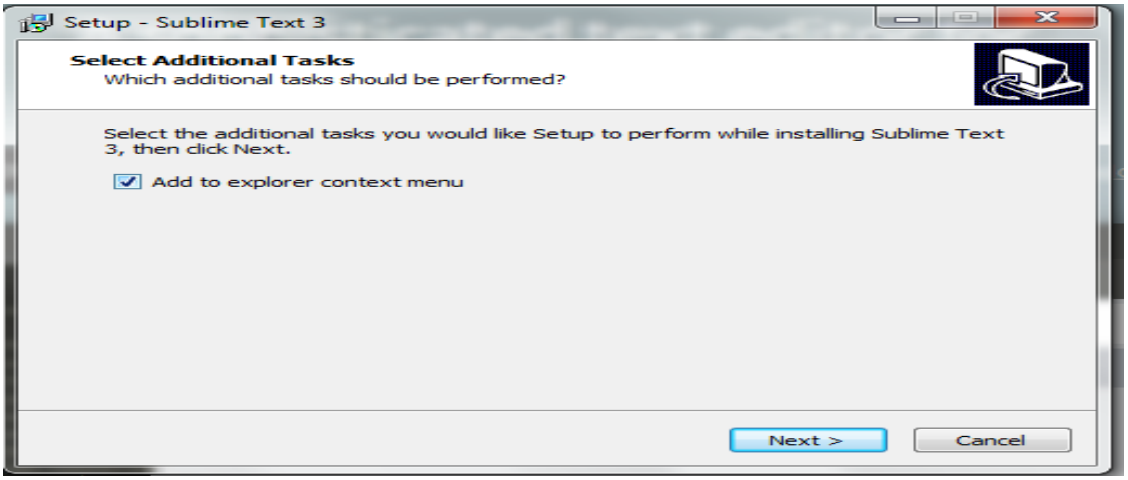
2. **Installation :**

Tout d'abord, comment obtenir Sublime Text ? Sublime Text est utilisable gratuitement et il est téléchargeable à l'adresse suivante : « https://www.sublimetext.com/ », Vous y trouverez la version téléchargeable pour les trois systèmes d'exploitation Windows, Mac et Linux et les différents Build disponibles avec ce qu'ils contiennent. Donc on le téléchargera en cliquant sur Download for Windows.

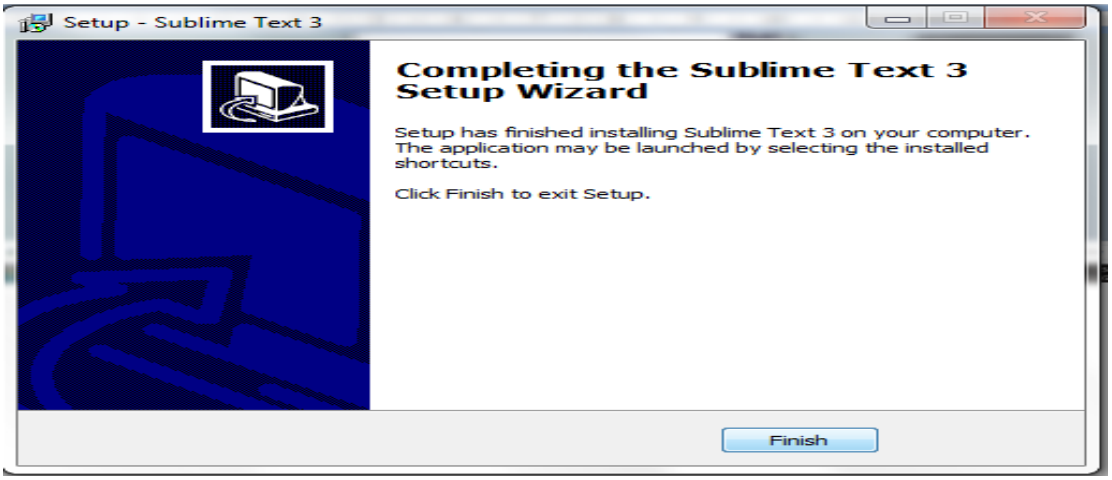


Donc après le téléchargement est terminé, accédez à nouveau à notre dossier de téléchargement et double-cliquez sur le fichier téléchargeable de Sublime Texte et lancer l'installation.





On peut ajouter un Sublime texte au menu contextuel quand on va faire clic droit sur un fichier html on pourra par exemple éditer grâce à Sublime. Puis clic **Next**>**Installe** et c'est fini.



Annexe 3 Installation du Postman API

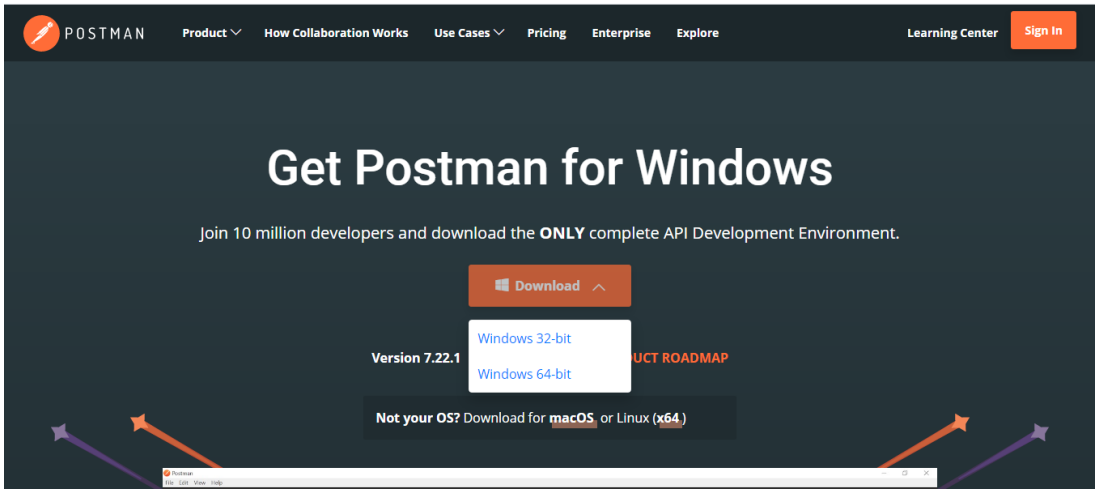
1. **Introduction :**

Postman est un outil de développement d'API (interface de programmation d'application) qui aide à créer, tester et modifier des API. Presque toutes les fonctionnalités dont tout développeur pourrait avoir besoin sont encapsulées dans cet outil. Il est utilisé par plus de 5 millions de développeurs chaque mois pour rendre leur développement d'API facile et simple. Il a la capacité de faire différents types de requêtes HTTP (GET, POST, PUT, PATCH), d'enregistrer des environnements pour une utilisation ultérieure, de convertir l'API en code pour différents langages (comme JavaScript, Python).

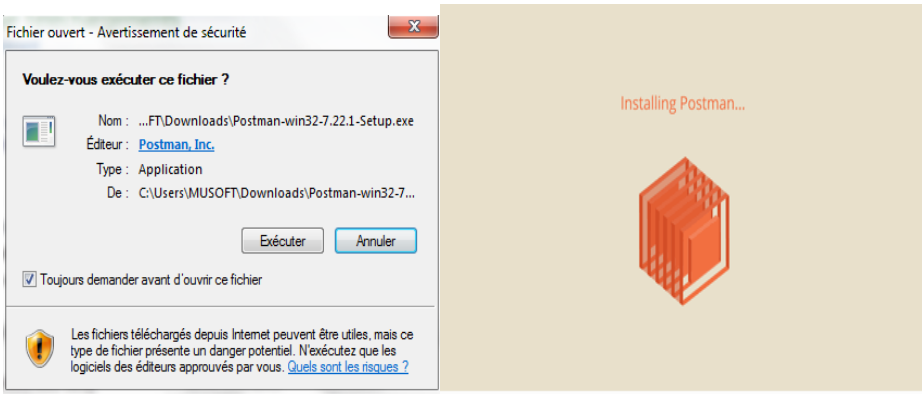
2. **Installation :**

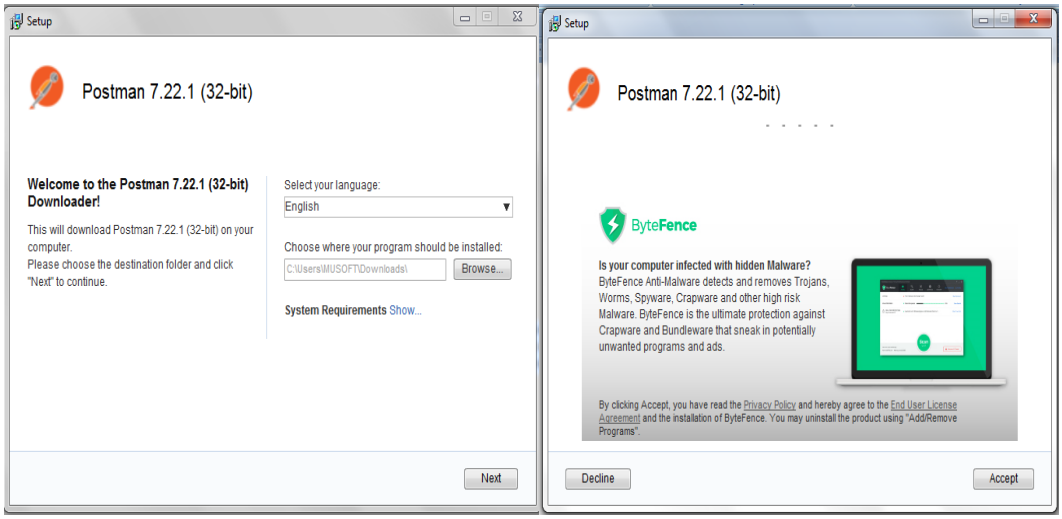
Tout d'abord, nous devons télécharger Postman à partir du site suivant : « https://www.postman.com/», et après en cliquant sur **Download the App** après

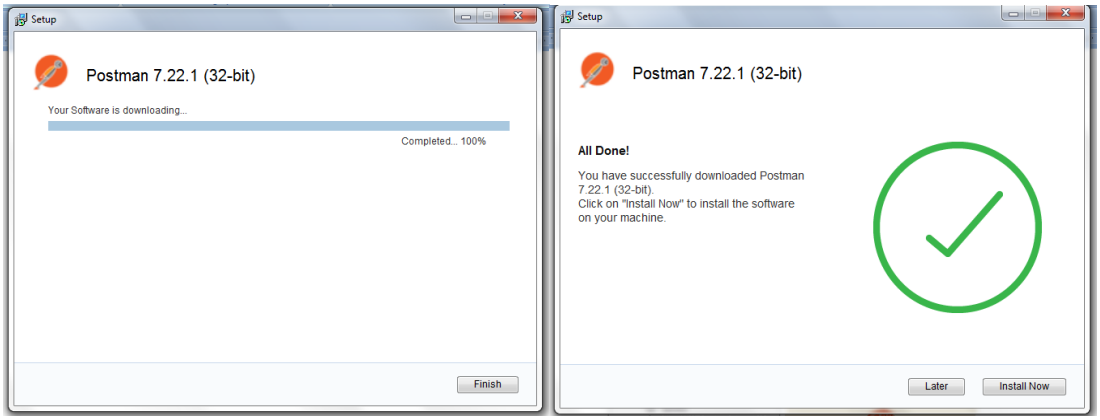
nous choisissons la version de notre Windows (32 ou 64) bits après téléchargement est en cours automatique.



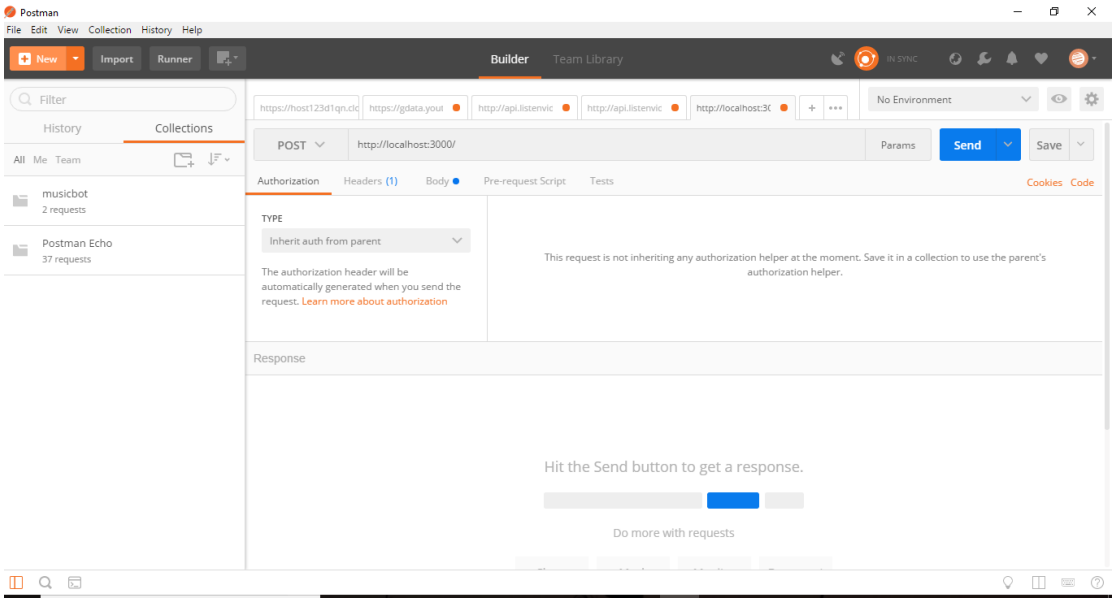
Après avoir téléchargé, on accède à notre dossier de téléchargement et double-cliquez sur le fichier téléchargeable de Postman et lancer l'installation en cliquons sur Exécuter > Next > Accept > Finish > Install Now







Après avoir téléchargé et installé le Postman, ouvrez le logiciel.



Postman interface.

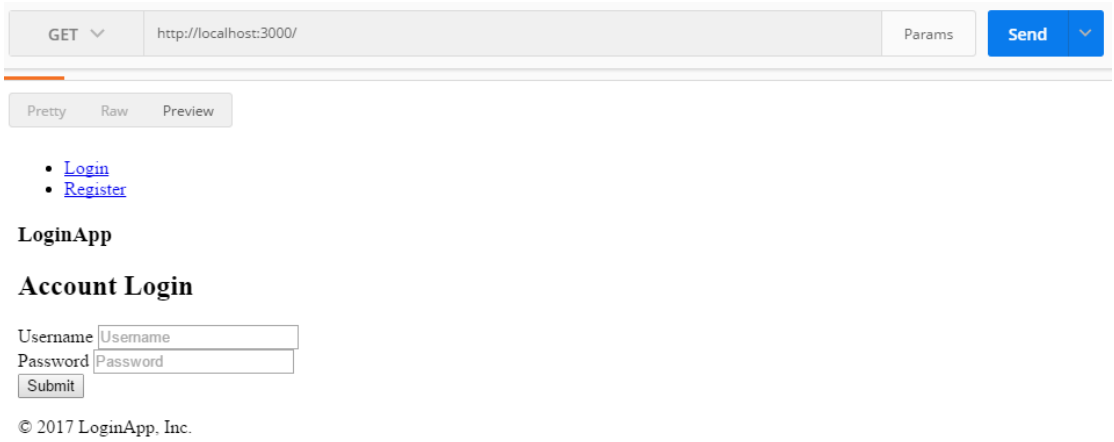
3. **Expliquer l'interface :**

- Le champ de saisie central le plus long qui ressemble à une barre de recherche est celui où l'URL que nous voulons GET ou POST ou DELETE, etc. est nourri.
- Juste à gauche, se trouve un bouton déroulant qui a toutes les différentes méthodes HTTP en option. Si vous souhaitez POSTER vers l'URL que vous avez spécifiée, sélectionnez POST.
- À droite se trouve le bouton params. Si vous cliquez dessus, une nouvelle interface apparaîtra. Les paramètres sont essentiellement les données que nous voulons envoyer au serveur avec notre demande. Nous utiliserons cette interface params pour POST pour mettre l'application un nouvel utilisateur.
- À gauche de ce bouton se trouve le bouton Envoyer qui est utilisé dans l'envoi de la demande au serveur ou à l'application dans ce cas.

Alors, commençons à envoyer et à recevoir des demandes via Postman.

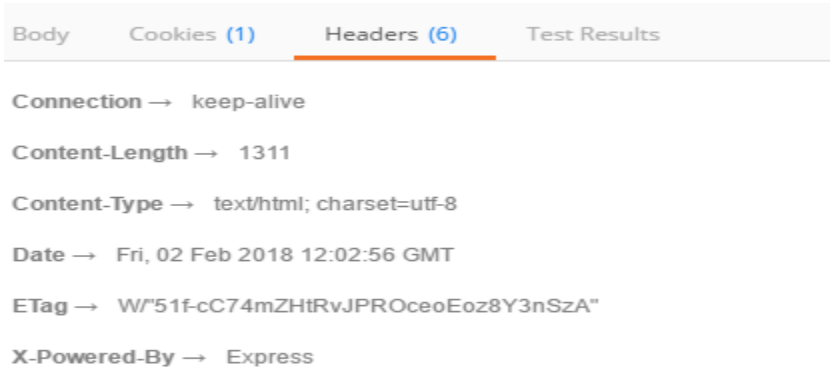
4. **Envoi et réception de demandes via Postman**

- Entrez l'URL que vous souhaitez frapper dans la barre d'URL que j'ai décrite ci-dessus. Je mettrai http: localhost: 3000 dans mon cas.
- Permet de sélectionner notre méthode HTTP pour envoyer la demande en tant que GET dans le bouton gauche. Cliquez maintenant sur le bouton Envoyer.



Obtenir localhost

- Vous serez renvoyé HTML de l'URL que vous OBTENEZ. J'ai sélectionné l'aperçu pour avoir un aspect de navigateur.
- Comme vous pouvez le voir dans la capture d'écran ci-dessous, avec la réponse du serveur ou de l'application, divers en-têtes sont également renvoyés avec la réponse principale.



Les en-têtes de retour obtiennent

5. **Explication de l'en-tête :**

Le premier en-tête renvoyé est **keep-alive**. Cela signifie essentiellement que la connexion du serveur avec l'utilisateur ne se tuera pas après un certain temps.

Content-length est la longueur du document html qui est retourné.

La **date** est l'heure à laquelle la demande a été faite au serveur pour retourner le fichier.

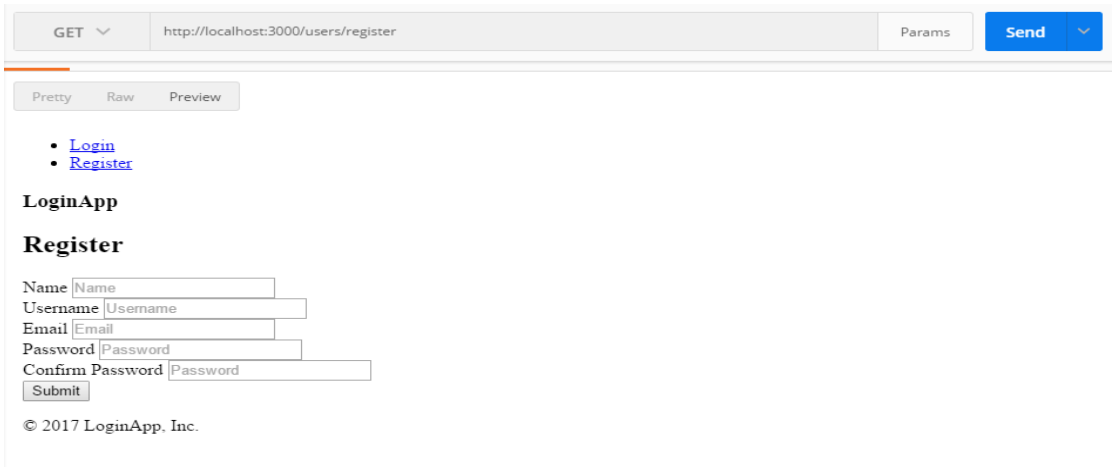
X-Powered-By envoie Express car le serveur d'application est Express.

Etag est un identifiant pour une version spécifique de la ressource. Cela permet d'économiser du temps et de la bande passante au cas où l'utilisateur demanderait à nouveau la même page sans aucune modification, alors le même fichier pourrait être envoyé.

6. Publication des données sur le serveur :

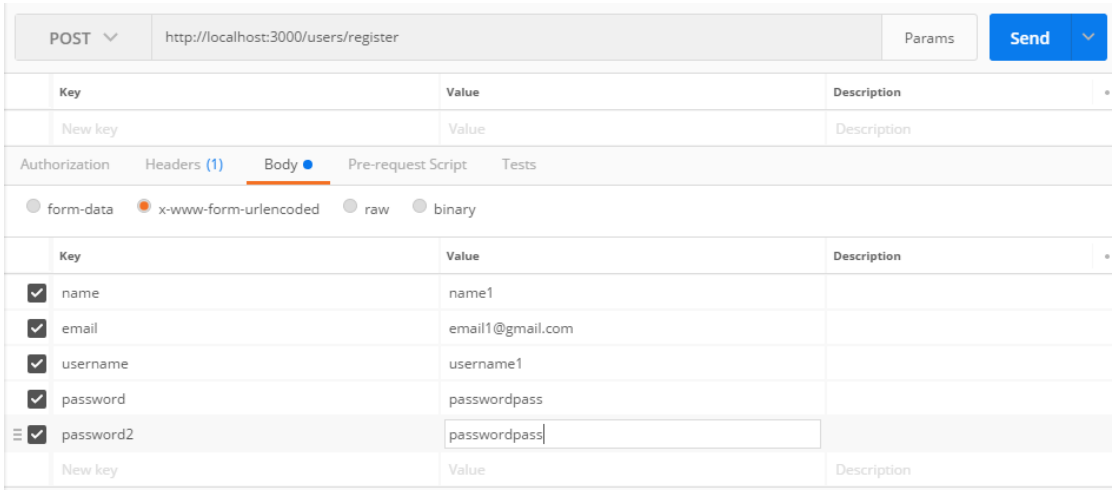
Maintenant, essayons de POSTER des données sur le serveur (serveur local).

Pour cela, nous allons d'abord OBTENIR le formulaire d'inscription.



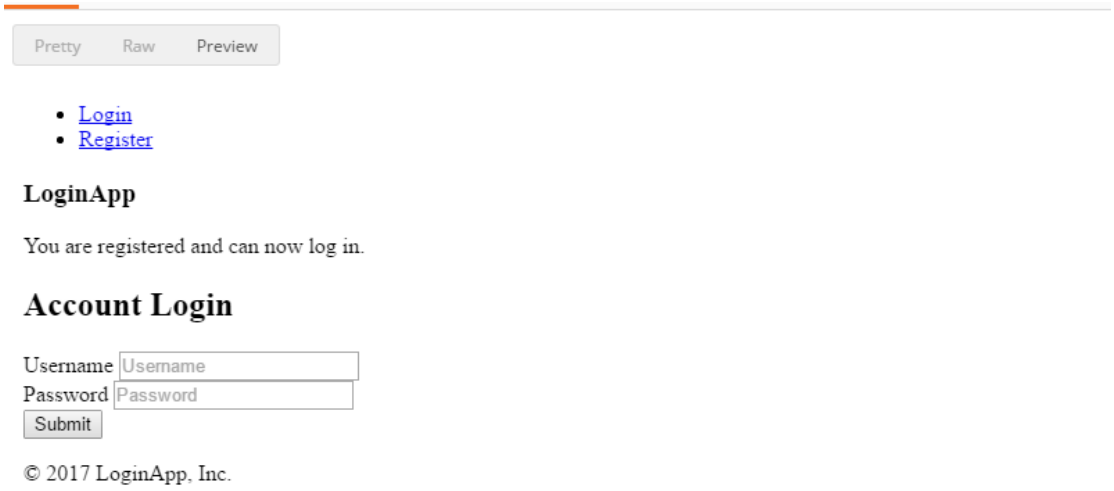
S'inscrire

- Modifiez la méthode HTTP de la prochaine requête que nous allons envoyer à POST. Ouvrez l'onglet Params du Postman. Cela nous aidera à envoyer le formulaire avec les valeurs souhaitées.



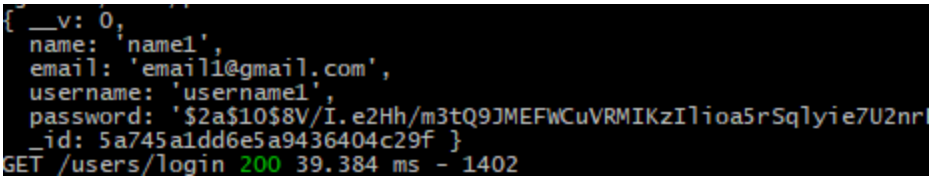
Valeur de formulaire remplie sous forme de pair clé-valeur dans l'onglet Paramètres du facteur.

- Après avoir appuyé sur Entrée, il POSTE le formulaire avec nos paires clé-valeur et renvoie la réponse.



Aperçu look Postman pour utilisateur enregistré

- Le terminal enregistre également l'utilisateur enregistré.



La console a enregistré l'utilisateur enregistré

Annexe 4 Node.js

1- Qu'est-ce que Node.js ?

Node.js est une plate-forme côté serveur basée sur le moteur JavaScript de Google Chrome pour créer facilement des applications réseau rapides et évolutives. Node.js utilise un modèle d'E / S non bloquant piloté par les événements qui le rend léger et efficace, parfait pour les applications en temps réel gourmandes en données qui s'exécutent sur des appareils distribués.

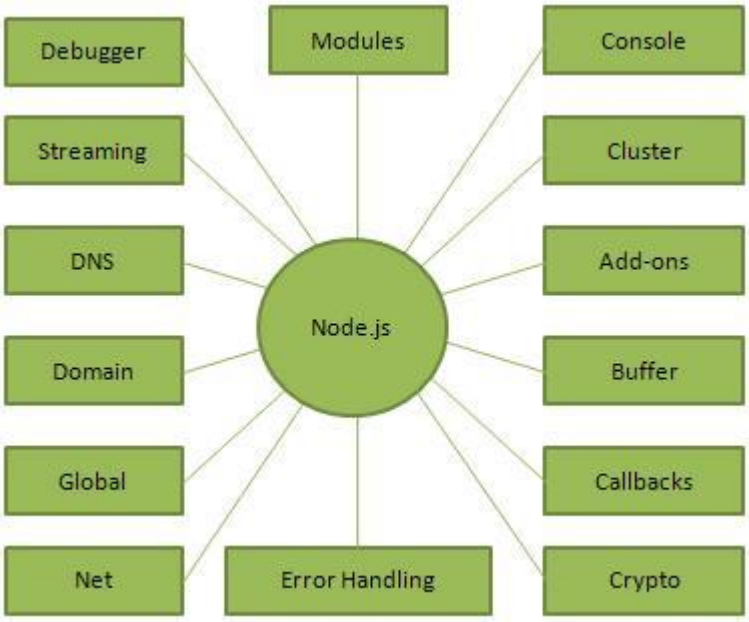
Node.js est un environnement d'exécution multiplateforme open source pour le développement d'applications côté serveur et de mise en réseau. Les applications Node.js sont écrites en JavaScript et peuvent être exécutées dans le runtime Node.js sur OS X, Microsoft Windows et Linux.

Node.js fournit également une riche bibliothèque de divers modules JavaScript qui simplifie dans une large mesure le développement d'applications Web utilisant Node.js.

Node.js = Runtime Environment + JavaScript Library

2- Concepts

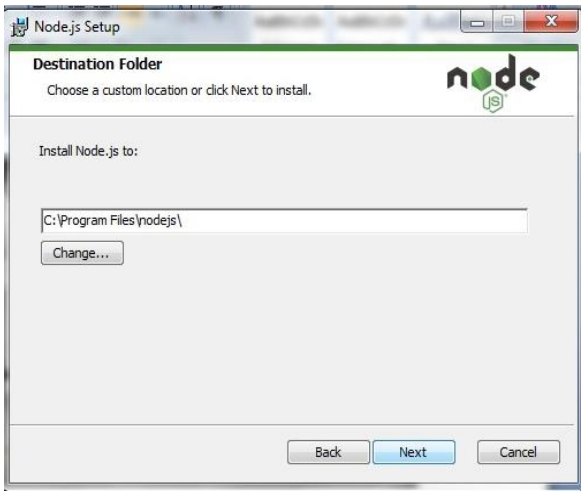
Le diagramme suivant décrit certaines parties importantes de Node.js

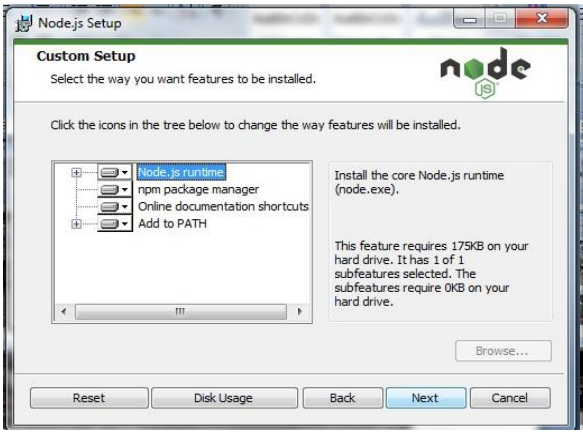


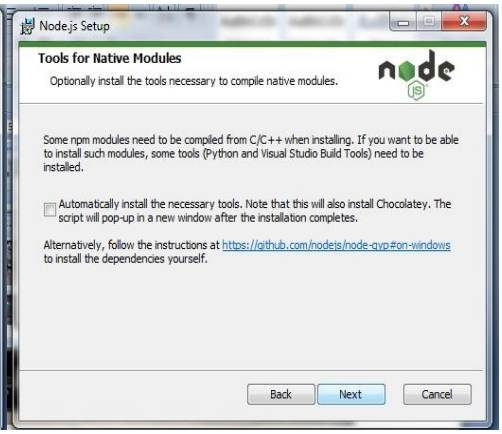
Télécharger l'archive Node.js

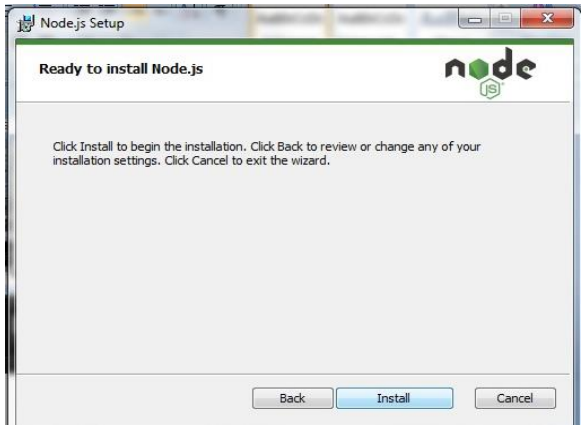
Télécharger la dernière version de Node.js sur lesite suivant : « https://nodejs.org/en/download/»

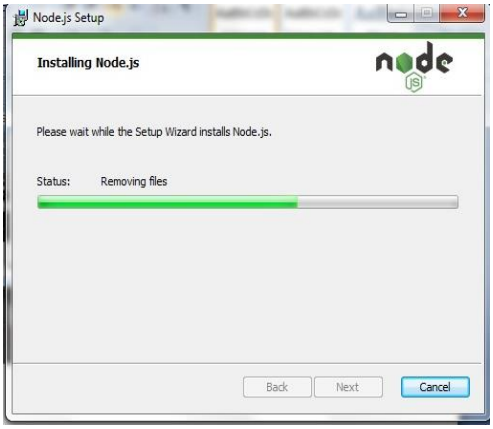
Utilisez le fichier MSI et suivez les invites pour installer Node.js. Par défaut, le programme d'installation utilise la distribution Node.js dans C: \ Program Files \ nodejs.

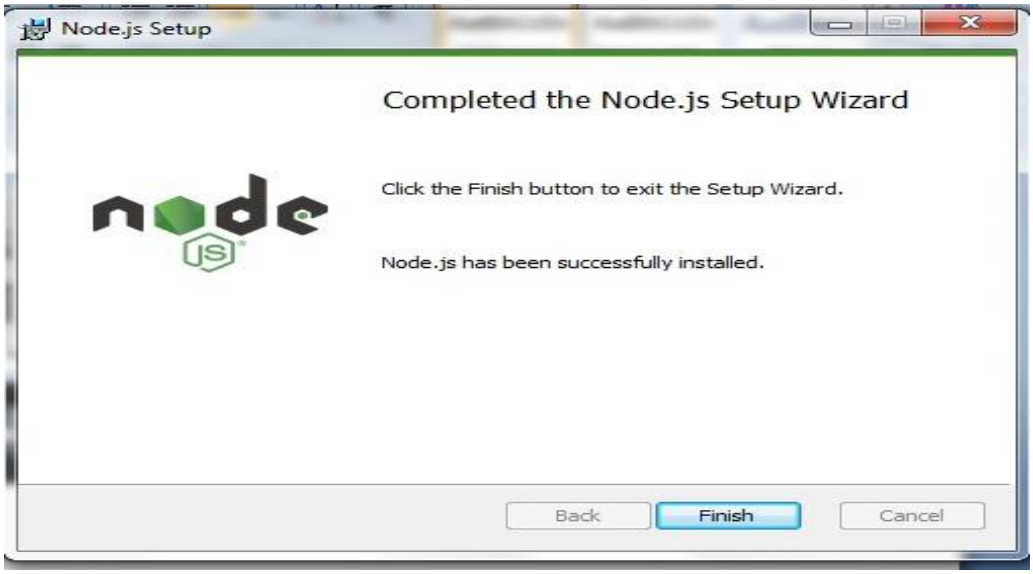












Vérifier l'installation : exécution d'un fichier

Créez un fichier js nommé **main.js** sur votre machine (Windows ou Linux) avec le code suivant.

```
/* Hello, World! program in node.js */
console.log("Hello, World!")
```

Maintenant, exécutez le fichier main.js en utilisant l'interpréteur Node.js pour voir le résultat

```
$ node main.js
```

Si tout va bien avec votre installation, cela devrait produire le résultat suivant : Hello, World!

Annexe 5 Cisco packet Tracer

1- Qu'est-ce que Cisco Packet Tracer ?

Cisco Packet Tracer est un outil de simulation qui vous permet d'utiliser des périphériques réseau Cisco avec un logiciel.
L'installation est gratuite pour tout le monde.
Il s'agit d'un outil très précieux pour les ingénieurs d'infrastructure qui peuvent acquérir une expérience de fonctionnement même s'ils n'ont aucune expérience de l'équipement réseau réel.

2- Que pouvez-vous faire ?

Vous pouvez placer un PC, un routeur, un commutateur L2, un commutateur L3, etc. sur le logiciel et les connecter pour créer un réseau.

Vous pouvez enregistrer le réseau créé et le distribuer à d'autres.

Vous pouvez essayer de créer une boucle ou de définir une combinaison qui provoque une erreur, ce qui ne devrait jamais être fait dans un vrai réseau.

Vous pouvez vérifier le comportement au moment de la panne, comme la façon dont le câble se déplacera lorsque vous le déconnecterez.

Si vous êtes en mesure de former des juniors, vous pouvez donner une tâche sans préparer une vraie machine.

Vous pouvez presque maîtriser les éléments de base de la couche 2 (couche liaison de données) et de la couche 3 (couche réseau), qui sont les sujets de base en tant qu'ingénieur réseau.

3- Que ne pouvez-vous pas faire ?

Vous ne pouvez pas vous connecter à des routeurs, des commutateurs ou des PC en dehors de Packet Tracer.

Seul un nombre limité de commandes Cisco peuvent être utilisées (90% peuvent être utilisées au niveau CCNA.)