Ministère de l'Enseignement Supérieur de la Recherche et de l'Innovation

Université Assane Seck de Ziguinchor
UFR Sciences et Technologies
Département Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique mention Informatique spécialité Génie Logiciel

Sujet :

Sécurité Blockchain : Étude analytique des aspects cryptographiques de la Blockchain

Présenté par : Mr IDRISSA BIAYE

Soutenance le 26/11/2022

Sous la direction de : Dr El-Hadj Malick NDOYE et Mr Malaw NDIAYE

Sous la supervision du Pr Ousmane DIALLO

Membres du jury:

Ousmane DIALLO	Professeur assimilé (MC CAMES)	Président	UASZ
El-Hadj Malick NDOYE	Maître Conférence titulaire (MA CAMES)	Encadrant	UASZ
Malaw NDIAYE	Assistant	Co-encadrant	UASZ
Ibrahima DIOP	Professeur assimilé (MC CAMES)	Examinateur	UASZ
Youssou FAYE	Professeur assimilé (MC CAMES)	Rapporteur	UASZ

Année : 2021 -2022

Remerciement

J'exprime ma profonde gratitude envers **ALLAH** le tout-puissant à qui nous devons soumission, obéissance et reconnaissance et prier sur le **prophète MOHANMED (PSL)** de m'avoir donné toute la volonté et le courage d'accomplir ce travail.

C'est avec un immense plaisir que j'exprime ma gratitude à ceux qui ont pu contribuer directement ou indirectement à l'élaboration de ce travail de mémoire.

Je tiens à remercier mes encadrants **Mr El-Hadj Malick NDOYE** et **Mr Malaw NDIAYE** d'avoir accepté de superviser et diriger ce travail. En plus des conseils avisés, de leur volonté d'aider et de leur investissement sans failles.

Je remercie le président du jury, le **Pr Ousmane DIALLO**, d'avoir supervisé ce travail et d'avoir accepté de présider le jury.

Je tiens également à remercier le **Pr Ibrahima DIOP** et le **Pr Youssou FAYE** qui a accepté d'être respectivement l'examinateur et le rapporteur de ce travail en vue de l'enrichir au travers leurs critiques et suggestions.

Nos vifs remerciements s'adressent aussi à tous les professeurs du département **d'informatique** qui nous ont accompagnés depuis le début de notre cursus universitaire. Plus particulièrement à **Pr Marie NDIAYE** qui est une maman pour nous.

J'adresse également mes remerciements à mes parents, frères et sœurs qui m'ont soutenu durant tout mon cursus scolaire.

J'adresse ma profonde gratitude à mes tuteurs **Ibrahima BA** et sa femme **Mariama TOURE**, ainsi qu'à la famille **NIAMADIO** plus particulièrement à ma tante **Kady BIAYE** et son **Mari**.

Je remercie également mes camarades de la première et deuxième promotion MPI, plus particulièrement les frères **Amadou SEYDI**, **Amadou NDIAYE, Ibrahima Soly SEYDI** et **Tida DJITTE** ainsi qu'à mes aînés qui m'ont apporté leur soutien et conseils à l'image des **Dr Souhaibou SAMBOU et Sidya DIENG**.

J'adresse ma profonde gratitude à mes camarades de classe qui sont devenu une seconde famille.

J'adresse également ma profonde gratitude aux personnes que j'ai eu l'honneur de côtoyer au sein de l'université Assane Seck de Ziguinchor et qui ont fini par occuper une place assez importante dans notre cœur.

C'est le moment aussi de remercier les membres de l'amical des étudiants ressortissants de la région de SÉDHIOU, dont je fus le président.

Dédicace

Je dédie ce travail :

- A mon père BOURAMA BIAYE et à ma MAMAN MARIAMA DIALLO,
- A mes frères ABDOURAHMANE BIAYE et OUSMANE TALL,
- A mes sœurs FATOUMATA BIAYE et SARATA BIAYE, MAME AWA BIAYE et SAUNA BIAYE,
- A mes grandes mères SARATA MANE SIDIBE et DIENABA SEYDI,
- A mes oncles paternels et maternels : SOULEYMANE SADIO, MAODO DIALLO, IBOU MANE, OUSMANE MANE et MOUSSA SADIO,
- Mes cousins et cousines : IBRAHIMA DIALLO et MARIAMA SADIO,
- A mon neveu MAMADOU LAMINE SY et mon grand frère DJIBY BIAYE,
- A mes neveux et nièces, cousins et cousines,
- Aux grandes familles DIALLO et BIAYE de SAMINE ESCALE,
- Aux grandes famille BIAYE et SADIO de NIAFOR,
- A mon défunt promotionnaire MAMOUR DIOUF,
- Au défunt aîné et camarade ALPHA SANE.

Résumé

La Blockchain a récemment émergé comme un nouveau courant de recherche, dans un large éventail de domaines avec plusieurs applications.

Elle fonctionne dans un environnement décentralisé, grâce à l'existence de plusieurs technologies de base telle que les signatures numériques, les hachages cryptographiques et les algorithmes de consensus distribués. Toutes les transactions sont effectuées de manière décentralisée afin d'éviter le fait qu'un élément central soit chargé de vérifier et de valider les opérations qui s'effectuent sur le réseau [1].

Par ailleurs, il est essentiel de résoudre ces problèmes de sécurité et de mettre en œuvre des fonctions de hachage plus efficaces pour garantir l'intégrité des données.

Une technologie particulièrement réussie est celle des smarts contracts.

Un smart contract est un ensemble de règles et de logique de programme de type scénario-réponse. C'est-à-dire qu'il s'agit d'un code partagé, décentralisé et de confiance déployée sur la blockchain. Les signataires du contrat doivent s'entendre sur les détails du contrat, les termes de la violation et la responsabilité en cas de rupture de contrat. Puis le déployer en tant que contrat intelligent sur la Blockchain pour exécuter automatiquement le contrat au nom des signataires. L'ensemble du processus est indépendant de toute autorité centrale.

Dans ce travail, nous mettons d'abord l'accent les fonctions de hachage et surtout les risques qui sont liés aux smartes contrats [2]. Dans la mesure où ces derniers ont la possibilité de stocker des données personnelles des utilisateurs et/ou des cryptomonnaies. Ces dernières années nous assistons à des pertes qui s'élèvent à des millions de dollars.

C'est pourquoi, nous proposons la mise en place de nouvelle architecture de vérification de la sécurité des smarts contracts à base de **machine learning**. Cette proposition rentre dans le cadre de renforcer la sécurité des smarts contracts afin d'éviter les pertes de données personnelles et des actives numériques dans un contexte décentralisé.

Par ailleurs, les fonctions de hachage qu'utilisent les plateformes Blockchain sont basées sur la construction Merkle-Damgard et la construction éponge. Il faut noter que les fonctions de hachage utilisées présentent de très bonnes caractéristiques en termes de complexité.

Abstract

Blockchain has recently emerged as a new research trend, in a wide range of fields with several applications. It operates in a decentralised environment, thanks to the existence of several core technologies such as digital signatures, cryptographic hashes and distributed consensus algorithms. All transactions are carried out in a decentralised manner in order to avoid the need for a central element to verify and validate the operations that take place on the network.

Furthermore, it is essential to solve these security problems and implement more efficient hash functions to guarantee data integrity. One particularly successful technology is that of smart contracts. A smart contract is a set of rules and program logic of the scenario-response type. That is, it is a shared, decentralised and trusted code deployed on the blockchain.

The signatories of the contract must agree on the details of the contract, the terms of breach and the liability for breach of contract. Then deploy it as a smart contract on the blockchain to automatically execute the contract on behalf of the signatories. The whole process is independent of any central authority.

In this work, we first focus on the hash functions and especially the risks that are associated with smart contracts. Insofar as these have the possibility to store personal data of users and/or cryptocurrencies. In recent years we have seen losses amounting to millions of dollars. This is why we propose the implementation of a new architecture to verify the security of smart contracts based on machine learning.

This proposal is in line with the objective of strengthening the security of smart contracts in order to avoid the loss of personal data and digital assets in a decentralised context.

Furthermore, the hash functions used by blockchain platforms are based on the Merkle-Damgard construction and the sponge construction. It should be noted that the hash functions used have very good characteristics in terms of complexity.

Table des matières

Liste des Tableaux

Liste des figures

Introduction Générale

Les organisations et entreprises sont confrontées à de nouveaux défis tels que la sécurité de l'information, la confiance et la transparence entre les différentes parties prenantes, la décentralisation du flux de travail, etc. C'est dans ce contexte que la technologie Blockchain est récemment apparue comme une tendance de recherche, avec des applications potentielles dans un large éventail d'industries et de contextes. En offrant de nouvelles opportunités de recherches afin de résoudre ces problèmes. Une technologie Blockchain particulièrement réussie est le smart contract, qui est largement utilisé dans les environnements commerciaux (par exemple : transactions financières de grande valeur). Ils sont activés si un certain nombre de conditions sont réunies. Ensuite, ils sont exécutés dans un réseau de nœuds qui ne se font pas confiance sans arbitrage par une autorité centrale [3]. Les smarts contracts facilitent la collaboration entre les entreprises, organisations ou les différentes parties prenantes dans le souci d'appliquer les conditions contractuelles auto-applicables sans l'intervention de tierce de confiance. Cependant, cela à des implications de sécurité en raison du potentiel de bénéficier financièrement d'un incident de sécurité (par exemple : identifier et exploiter une vulnérabilité dans le smart contract ou sa mise en œuvre).

Ces dernières années nous avons constatées un flamber des attaques contre les plateformes Blockchain qui intègrent les smarts contracts. Les attaquants ont pu dérober des millions de dollars et dans certains cas ont pu avoir accès à des données personnelles des utilisateurs.

C'est pourquoi les enjeux de sécurité de ces derniers doivent être fondamentaux. C'est dans ce contexte que s'inscrit notre étude qui évalue d'abord le niveau de sécurité de la technologie blockchain. Pour ce faire nous allons voir les notions de sécurité qui interviennent dans la technologie blockchain [4]. Ensuite nous allons effectuer l'état de l'art afin de mettre en évidence comment ces concepts interagissent dans la technologie. Enfin nous terminerons par une discussion et perspective.

Un accent particulier sera mis sur les fonctions de hachage et les smarts contracts. Les fonctions de hachage de par leur rôle indispensable dans tous les processus de la technologie Blockchain. Elles influencent le passage du langage de haut niveau (compréhensible par l'homme), au langage

de bas niveau (compréhensible par la machine). Alors que pour les smarts contracts, c'est de par leur implication à contenir des données et des crypto-monnaies.

Chapitre I : Notions de sécurité informatique pour la technologie blockchain

Notions de sécurité pour la Blockchain

La Blockchain offre une approche innovante pour stocker des informations, exécuter des transactions, exécuter des fonctions et établir la confiance dans un environnement ouvert. Beaucoup considèrent la Blockchain comme une percée technologique pour la cryptographie et la cyber sécurité, avec des cas d'utilisation allant des systèmes de crypto-monnaie déployés à l'échelle mondiale comme le Bitcoin, aux smarts contracts etc. Bien que la Blockchain ait suscité un intérêt croissant, la cryptographie derrière qui permet d'assurer la sécurité qui sous-tend cette nouvelle technologie continue d'être au centre des débats.

La sécurité informatique couvre l'ensemble des moyens informatiques mis en œuvre dans le but de réduire le vol ou la fuite d'informations, la modification des données, l'échec des services et assurer que les ressources sont utilisées comme prévu. Il se compose d'un certain nombre de mécanismes qui fournissent divers services de sécurité permettant d'atteindre un certain niveau de protection. Ces mécanismes empêchent la manipulation malveillante du système par des personnes non autorisées et empêchent les utilisateurs d'effectuer des actions inattendues qui pourraient endommager l'ensemble du système. Ils garantissent également le fonctionnement permanent du service et peuvent réduire la défaillance du système.

C'est dans ce contexte que nous verrons d'abord les mécanismes de cryptographie qui permettent d'assurer la sécurité de la technologie Blockchain. Ensuite nous parlerons de la problématique de la sécurité informatique et enfin des services de sécurité.

1 Les mécanismes de sécurité informatique

La cryptographie est sans doute le mécanisme de sécurité le plus utilisé dans le contexte de l'informatique pour assurer la sécurité des systèmes, des réseaux informatiques et plus particulièrement des réseaux Blockchains. C'est pourquoi il est difficile de réduire ce vaste domaine en une simple définition, de ce fait une compréhension de la cryptographie ne pourra s'acquérir qu'au fur et à mesure d'étudier ses fondements.

1.1 Cryptographie symétrique

La cryptographie symétrique fait référence au processus d'utilisation d'une clé unique pour le chiffrement et le déchiffrement [5] [6]. Cela signifie que la même clé doit être disponible pour plusieurs personnes si elles souhaitent échanger des messages en utilisant cette forme de cryptographie. Le processus de chiffrement symétrique est décrit comme suit :

Les algorithmes symétriques utilisent la même clé pré-partagée pour chiffrer et déchiffrer les données ; une méthode également connue sous le nom de chiffrement par clé privée.

Par exemple, Alice et Bob résident dans des villes différentes et souhaitent s'échanger des messages secrets par courrier. Alice souhaite faire parvenir un message secret à Bob.

Le chiffrement par clé privée utilise un algorithme symétrique. Alice et Bob possèdent des clés identiques pour ouvrir un même cadenas. L'échange de clés a eu lieu avant l'envoi des messages secrets. Alice écrit un message secret et le place dans une petite boîte qu'elle verrouille à l'aide du cadenas. Elle envoie la boîte à Bob. Pendant le transfert, le message est en sécurité dans la boîte. Lorsque Bob reçoit la boîte, il utilise sa clé pour ouvrir le cadenas et récupérer le message. Bob peut alors réutiliser la boîte et le cadenas pour envoyer un message secret à Alice tel que décrit dans [7].

Si Bob désire communiquer avec Carol, il a besoin d'une nouvelle clé pré-partagée pour que cette communication ne soit pas révélée à Alice. Plus le nombre de personnes avec lesquelles Bob souhaite communiquer de manière sécurisée sera élevé, plus le nombre de clés à gérer sera important.

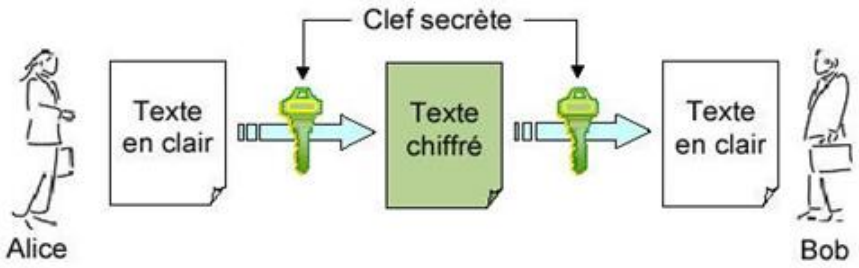


Figure 1 : Utilisation de la même clé pour le chiffrement symétrique

1.1.1 Le chiffrement par bloc

Le chiffrement par bloc transforme un bloc de texte en clair d'une longueur fixe en bloc de texte crypté de 64 ou 128 bits. La taille du bloc correspond à la quantité de données chiffrées à un moment donné. Pour déchiffrer ce texte crypté, appliquez la transformation inverse au bloc de texte crypté en utilisant la même clé secrète [8].

En règle générale, le chiffrement par bloc génère des données de sortie plus volumineuses que les données d'entrée, car le texte chiffré doit être un multiple de la taille du bloc [9]. DES (Data Encryption Standard) et AES (Advanced Encryption standard) sont les plus connus parmi les algorithmes de chiffrement par bloc [10] [11], par exemple, est un algorithme symétrique qui chiffre les blocs en segments de 64 bits à l'aide d'une clé de 56 bits. Pour ce faire, l'algorithme prélève les données par segment (des segments de 8 octets, par exemple), jusqu'à ce que tout le bloc soit rempli. Si la quantité de données d'entrée est inférieure à un bloc complet, l'algorithme ajoute des données artificielles, ou des blancs, jusqu'à ce que les 64 bits soient utilisés

1.1.2 Le chiffrement par flux

Contrairement au chiffrement par bloc, le chiffrement de flux permet de chiffrer du texte en clair, à raison d'un bit à la fois. Le chiffrement de flux correspond à un chiffrement par bloc avec une taille de bloc d'un seul bit [12]. Avec le chiffrement de flux, la transformation de ses plus petites unités de texte en clair dépend de leur position dans le processus de chiffrement. Le chiffrement de flux peut se révéler beaucoup plus rapide que le chiffrement par bloc. De plus, cette méthode n'entraîne pas d'augmentation de la taille du bloc, car elle peut chiffrer un nombre arbitraire de bits [13].

A5/1 est un chiffrement de flux qui assure la confidentialité des communications vocales et chiffre les communications par téléphones mobiles. Il est également possible d'utiliser DES en mode de chiffrement de flux.
A titre d'exemple aussi nous pouvons citer RC4.
Les systèmes cryptographiques complexes peuvent combiner les modes de chiffrement de flux et par bloc au sein d'un même processus.

1.2 Cryptographie asymétrique

Elle fait référence au processus d'utilisation de deux clés pour le chiffrement et le déchiffrement. N'importe quelle clé peut être utilisée pour le chiffrement et le déchiffrement. Le chiffrement des messages avec une clé publique peut être déchiffré à l'aide d'une clé privée et les messages chiffrés par une clé privée peuvent être déchiffrés à l'aide d'une clé publique. Comprenons cela à l'aide d'un exemple [14]. Bob utilise la clé publique d'Alice pour chiffrer les messages et l'envoie à Alice. Alice peut utiliser sa clé privée pour déchiffrer le message et en extraire le contenu. Les messages chiffrés avec la clé publique d'Alice ne peuvent être déchiffrés que par Alice car elle seule détient sa clé privée et personne d'autre. C'est le cas d'utilisation générale des clés asymétriques. Il y a une autre utilisation que nous verrons en discutant des signatures numériques.

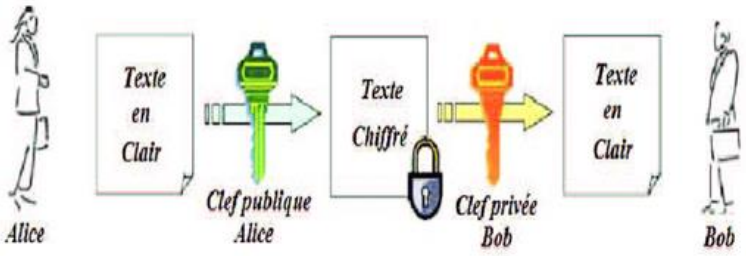


Figure 2: Utilisation de deux clés distinctes pour le chiffrement et le déchiffrement asymétrique

A titre d'exemple d'algorithmes asymétriques nous pouvons citer : RSA, El-Gamal, Robin Micheal, Goldwasser Micali etc.
La force de sécurité de ces algorithmes repose sur **les fonctions à sens unique** qu'elles contiennent et qui sont irréversibles. Par exemple, le RSA repose sur le problème de factorisation des nombres entiers alors que Diffie-Hellman et El-Gamal repose sur le problème du logarithme discret.
A cela s'ajoute le problème du logarithme discret elliptique sur lequel repose les systèmes ou algorithmes qui interviennent dans la sécurité et mécanisme qui sont basés sur ce dernier.

1.3 Les fonctions de hachages

Les fonctions de hachage sont considérées comme des éléments-clés de tous les protocoles cryptographiques, ainsi que dans de nombreuses applications de sécurité. Elles permettent d'assurer l'intégrité des données (afin de détecter les modifications non autorisées dans les fichiers), le stockage de mot de passe, les codes d'authentification et la génération de mots de passe aléatoire. Les fonctions de hachage jouent un rôle fondamental dans la technologie Blockchain de par leur propriété unidirectionnelle. Elles interviennent presque exclusivement à tous les niveaux. De la génération des adresses de compte, les signatures numériques, le hachage de transaction, le hachage d'état et le hachage d'en tête de bloc.

Plusieurs algorithmes de fonctions de hachage ont été proposés parmi lesquels nous pouvons citer : Blake-256, SHA-2(224,256, 524), Keccak et SHA-3 [15].

Elles prennent des entrées de longueur arbitraire (également appelée message ou texte plat), pour générer une sortie de longueur fixe (appelée condensé de message ou code de hachage).

A l'exception de la famille de fonction de hachages Keccak qui prend en entrée des données de taille arbitraires, pour produire une sortie de taille arbitraire [16].

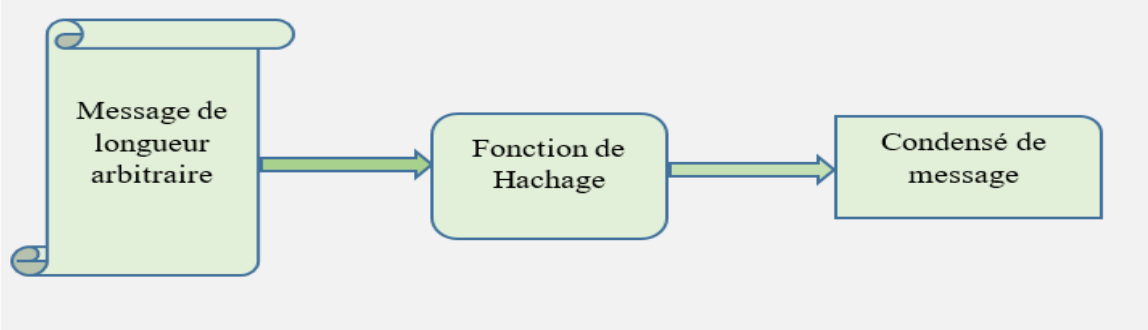


Figure 3 : Hachage des données

La définition mathématique d'une fonction de hachage (H) est donnée comme suit :

Soit Σ l'alphabet tel que Σ = {0, 1}.

$$H : \{0, 1\}^* \rightarrow \{0, 1\}$$

Où, {0, 1}* fait référence à l'ensemble des éléments binaires de toute longueur, y compris la chaîne vide. Parallèlement, {0, 1} n est utilisé pour désigner un ensemble d'éléments binaires de longueur n. Ainsi, un ensemble d'éléments binaires de longueur arbitraire, est converti en ensemble d'éléments binaires de longueur fixe à l'aide de la fonction de hachage (à l'exception de la fonction Keccak).

1.3.1 La famille des SHA-2 (224-256-384-512)

La famille des fonctions SHA-2 est également connue sous le nom d'algorithme de hachage sécurisé, qui représente une série de fonctions de hachage mises en œuvre par la National Security Agency des États-Unis [17]. Veuillez noter que SHA-2 a été créé à l'origine sur la base de SHA-0 et SHA-1, il représente donc la suite logique de ces algorithmes. Quand on parle de SHA-256, il s'agit en fait de la signature du fichier de données. Par exemple, SHA-256 peut générer une signature de 32 octets (soit 256 bits).

Cependant, il est important de savoir que le hachage ne permet pas de crypter les données, mais produit plutôt une signature ou une empreinte. Cela s'avère être très pratique dans la mesure où l'empreinte est facilement reconnaissable ou identifiable. En termes de performance, nous précisons que l'algorithme SHA-2 n'est pas cassable jusqu'à présent [18].

1.3.2 La fonction Keccak

Keccak a été publié par Guido Bertoni, Joan D'Ameen, Michel el Peeters et Gilles Van Assche. La construction de la fonction Keccak est dite dans l'éponge, qui est une construction itérative simple d'une fonction de sortie de longueur variable basée sur un arrangement de longueur fixe (ou transformation) d'entrée de longueur variable [19]. Il se compose de phases d'absorption qui utilisent des blocs de message pour mettre à jour l'état interne et produit une compression de sortie. Les deux étapes impliquent des fonctions de permutation composées d'instruction logique. La conception de la série Keccak offre une grande flexibilité. Deux paramètres à savoir le rapport r et le volume c définissent les membres de la famille des éponges Keccak [20].

Le ratio définit la taille du bloc, la somme r + c doit être égale à 1600, définit la taille de l'état et ainsi que la taille des permutations.

La série de hachage de fonction Keccak convient au système de 64 bits. Cependant il existe une astuce appelée consultation de table qui peut être utilisée pour répartir l'état interne de 64 bits entrelacés en bits de deux mots de 32 bits. Le nombre de visites à table est maintenu dans une fourchette raisonnable. Par conséquent l'implémentation 32 bits présente de bonnes performances [21] .

1.3.3 La fonction de hachage BLAKE-2

BLAKE-2 est une fonction de hachage cryptographique qui est plus rapide que MD5, SHA-1, SHA-2 et SHA-3, tout en étant au moins aussi sûr que la dernière norme SHA-3. BLAKE2 a été adopté par de nombreux projets en raison de sa grande vitesse, de sa sécurité et de sa simplicité.
Avec les autres finalistes, BLAKE est supposé être une fonction de hachage très puissante [22]. Même s'il n'a pas été retenu comme vainqueur, il bénéficie d'une large marge de sécurité, de très bonnes performances en logiciel, et a attiré une quantité considérable de cryptanalyse [23] [24].

1.3.4 La fonction SHA-3

La norme spécifie la série de fonctions Secure Hash Algorithmes 3 (SHA-3) pour les données binaires. Chaque fonction SHA-3 est basée sur un exemple de l'algorithme Keccak sélectionné par le NIST [25] comme gagnant du concours d'algorithme de hachage cryptographique SHA-3. La norme stipule également la série de calculs mathématiques Keccak-P permutation [26] [27], y compris les permutations de Keccak, pour promouvoir le développement de fonctions cryptographiques supplémentaires basées sur la permutation. La série SHA-3 contient quatre fonctions de hachage cryptographique, appelées SHA3-224, SHA3-256, SHA3-384, SHA3-512, et deux fonctions de sortie extensible (XOF), appelé SHAKE-128 et SHAKE-256 tel que précisé dans [28]. La fonction de hachage fait partie intégrante de nombreuses applications importantes de sécurité de l'information. Notamment la génération et la vérification de signature numérique, dérivation de clé et la génération de mots pseudo-aléatoire. La fonction de hachage spécifiée dans cette norme complète la fonction de hachage SHA-1 et la série de fonction de hachage SHA-2.
Ces fonctions doivent satisfaire un certain nombre de propriétés face aux types d'attaques génériques. A savoir l'attaque par collision, par pré image et seconde pré image.

1.3.5 Description des types d'attaque générique à l'endroit des fonctions de hachages

1.3.5.1 Attaque par collision

Il est impossible pour un attaquant de trouver la même valeur de hachage pour deux messages distincts. Lorsqu'un pair de message différent produit le même hash, alors une attaque par collision se produit. La fonction de hachage doit avoir la particularité de ne pas produire la même valeur de hachage pour différents messages.

1.3.5.2 Résistance aux attaques par pré-image

La pré-image est le message haché à une valeur donnée. Dans les attaques de pré-image, on suppose généralement qu'il y a au moins un message haché à une valeur donnée. En d'autres termes, l'attaquant doit découvrir qu'il est impossible d'obtenir les données d'origine à partir de la valeur de hachage donnée [29].

1.3.5.3 Résistance aux attaques par second pré-image

La deuxième pré-image renvoie un message avec la même valeur que le message donné (c'est-à-dire sélectionné au hasard), appelé la première pré-image. Evidemment, la deuxième pré-image doit être différente de la première. Ici, nous supposons que l'attaquant a également obtenu la valeur de hachage de la première pré-image. Sinon l'attaquant peut calculer par lui-même. Dans ce dernier cas, le coût du hachage de la première pré-image est également à la charge de l'attaquant. Vous pouvez également utiliser des attaques de pré-image par force brute pour trouver une deuxième pré-image. On ignore juste la première pré-image, sauf qu'on peut faire attention à ne pas essayer le même message que la première pré-image. En sélectionnant aléatoirement des messages, en supposant que le domaine de la fonction de hachage soit beaucoup plus grand, la probabilité que la deuxième pré-image soit égale à la première pré-image est négligeable donc cette possibilité est généralement ignorée.

1.3.6 Avantages et vulnérabilité des fonctions de hachage

Le tableau ci-dessous présente quelques avantages et inconvénients des fonctions de hachages qui ont été énumérées :

Fonction de hachages	Avantages	Vulnérabilités
SHA-2(224, 256, 384, 512)	• Propose un équilibre entre espace de stockage et sécurité, • Unilatérale, • Résistance aux attaques classiques	Occupe le processeur et rend les bases de données des identifiants/mot de passe plus volumineuses.

Keccak	• Unilatérale, • Résistantes aux attaques classiques	Pour l'instant, aucune vulnérabilité détectée.
BLAKE-2	• Très rapide en termes de vitesse, • Unilatérale	Moins sécurisée que ces pères
SHA-3	• Unilatérale, • Résistance aux attaques classiques	Ajoute une porte dérobée à l'algorithme de Keccak. Mais pour l'instant aucune vulnérabilité n'a été détectée.

Tableau 1 : Avantages et inconvénients des fonctions de hachages

1.3.7 Etude comparative des fonctions de hachages

Premièrement, les familles de fonctions de hachages SHA-2 et Keccak sont basées sur des principes de conception très différents. S'il y a une percée majeure dans la cryptanalyse pour l'une des fonctions de hachage (et cela devra être une grande percée), alors l'attaque ne sera probablement pas appliquée à l'autre.

Deuxièmement, la famille SHA-2 et Keccak ont des caractéristiques de mise en œuvre différentes. Ainsi, pour une application donnée, il peut être intéressant de pouvoir choisir l'algorithme qui présente le comportement le plus favorable pour une plateforme donnée.

Par exemple, le Keccak est plus convivial pour le matériel et plus adapté aux applications embarquées ou la puissance et le coût est limité.

Il faut noter cependant que la fonction BLAKE-2 est plus rapide en matière de vitesse de calcul que les fonctions SHA-2 et Keccak. En même temps, elle est moins sécurisée que ces derniers. C'est ce qui justifie peut-être la combinaison de deux fonctions de hachage dans la machine virtuelle de Tezos.

Nous rappelons que ces deux fonctions ne sont pas toujours cassables. Ainsi pour les types d'attaque classique qui ont été énumérés ci-dessus, la complexité de ces fonctions s'écrie de la manière suivante :

Pour les attaques par collision la complexité [30] est de la forme $\mathbf{O(n) = 2^{n/2}}$, pour les attaques de pré-image et de second pré-image la complexité sest de la forme $\mathbf{O(n) = 2^{n}}$.

Attaques génériques	Complexités
Collision	$\mathbf{2^{n/2}}$
Pré-image	$\mathbf{2^{n}}$
Second pré-image	$\mathbf{2^{n}}$

Tableau 2 : Récapitulatif de la complexité des attaques génériques

Par conséquent, nous pouvons dire que ces fonctions de hachages présentent de bonne performance, une résistance aux attaques générique avec une complexité remarquable. Ce qui nous conforte dans l'idée de dire que la technologie Blockchain est quasi infaillible.
C'est dans ce sillage que nous allons aborder la notion d'arbre de Merkle.

1.4 Racine de Merkle, arbre de Merkle et trie de Merkle Patricia

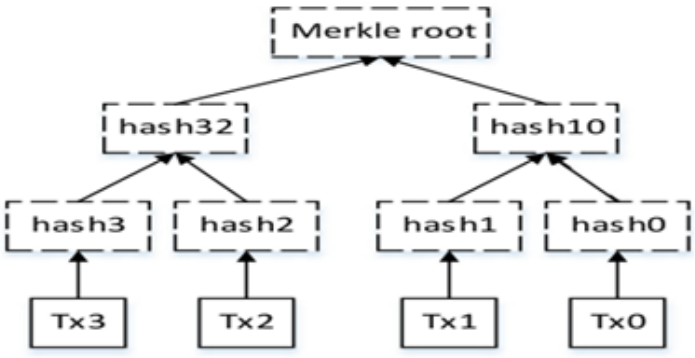


Figure 4 : Arbre de Merkle

Dans une Blockchain, toutes les transactions d'un bloc sont représentées par un seul hachage appelé racine de Merkle [31], qui est stocké dans l'en-tête du bloc. La racine de Merkle est la dernière

valeur de hachage de l'arbre de Merkle construit à partir des hachages des transactions d'un bloc. La *Figure 4* montre la construction d'un arbre de Merkle à partir d'un exemple de bloc comportant quatre transactions. Même si un bloc dans une Blockchain peut contenir quelques milliers de transactions. Pour calculer la racine de Merkle, les hachages des transactions d'un bloc sont hachés par paires. Les résultats de hachage sont continuellement appariés et hachés à nouveau jusqu'à l'obtention de la racine de Merkle qui est le dernier résultat de hachage. La modification de toute donnée de transaction sera détectée puisque la racine Merkle des transactions modifiées sera différente de la racine Merkle précédemment stockée dans l'en-tête du bloc. Les nœuds de vérification simplifiée des paiements (SPV) utilisent la racine Merkle pour vérifier si une transaction de paiement déclarée se trouve ou non dans la Blockchain. Le nœud SPV ne stocke que les en-têtes de bloc, laissant les données des transactions. Après avoir reçu la demande de transaction de paiement, le nœud SPV demande les branches Merkle du hachage de la transaction à un serveur Blockchain.

Le nœud SPV calcule ensuite la racine de Merkle à partir du hachage de la transaction et des branches de Merkle reçues du serveur. Si le résultat est le même que la racine Merkle déjà stockée dans l'en-tête du bloc, la transaction est dans la Blockchain et donc acceptée. Par exemple, pour vérifier si Tx0 sur la *Figure 4* est bien dans la Blockchain, le nœud SPV demande seulement $hash1$ et $hash32$ (branches Merkle de $hash0$). A l'aide des $hash0$, $hash1$, et $hash32$, le nœud SPV calcule la racine de Merkle et la compare à la racine de Merkle stockée dans l'en-tête du bloc. Par conséquent, le nœud SPV peut vérifier si les transactions sont dans la Blockchain sans avoir à télécharger toutes les transactions du bloc (Tx1, Tx2 et Tx3 dans ce cas). Dans Ethereum, chaque nœud complet stocke l'état global du réseau. L'état global contient des informations telles que le solde du compte, le prix du gaz, et la limite de gaz pour tous les comptes du réseau. Nous allons revenir sur ces notions dans le chapitre suivant.

1.5 Signature numérique

Plus tôt, nous avons discuté de la cryptographie à l'aide de clés asymétriques. Les cas les plus importants pour l'utilisation de clés asymétriques sont : la création et la vérification d'une signature numérique. Les signatures numériques sont très similaires à une signature faite par un individu sur un morceau de papier. Une signature numérique aide à identifier un individu. Cela permet également de s'assurer que les messages ne sont pas falsifiés en transit. Comprenons les signatures

numériques à l'aide d'un exemple. Alice veut envoyer un message à Bob. Comment Bob peut-il identifier et s'assurer que le message vient d'Alice uniquement et qu'il n'a pas été modifié ou falsifié en transit ? Au lieu d'envoyer un message/transaction brut, Alice crée un hachage de l'intégralité de la charge utile et crypte le hachage avec sa clé privée. Elle ajoute la signature numérique résultante au hachage et la transmet à Bob. Lorsque la transaction atteint Bob, il extrait la signature numérique et la déchiffre à l'aide de la clé publique d'Alice pour trouver le hachage d'origine. Il extrait également le hachage d'origine du reste du message et compare les deux hachages. Si les hachages correspondent, cela signifie qu'il provient en fait d'Alice et qu'il n'a pas été falsifié [32].

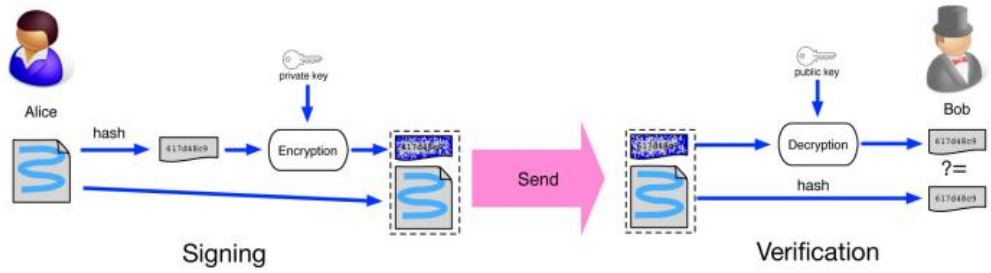


Figure 5 : Description du déroulement d'une signature numérique *[33]*

Les signatures numériques sont utilisées pour signer les données de transaction par le propriétaire de l'actif (c'est-à-dire le crypto monnaie).

Il existe différents algorithmes qui sont utilisés pour effectuer la signature numérique.

1.5.1 Description et comparaison des algorithmes de signature numérique

Les trois algorithmes de signature numérique les plus courants sont DSA (Digital Signature Algorithm), RSA (Rivest-Shamir-Adleman) et ECDSA (Elliptic Curve Digital Signature Algorithm). Comme vous pouvez le constater, les algorithmes ECDSA et ECDH (Elliptic Curve Deffi-Hellman) sont un cryptosystème des ECC (développer dans la section ci-dessous) utilisé dans la technologie Blockchain. Nous allons développer dans la suite en détail le fonctionnement de cet algorithme. Par ailleurs, tous quatre génèrent et vérifient les signatures numériques. Ces algorithmes dépendent des techniques de chiffrement asymétrique et de clé publique. Dans le cas des signatures numériques, deux opérations sont requises :

- Génération de clés;
- Vérification de la clé.

Ces deux opérations nécessitent le chiffrement et le déchiffrement de la clé.

L'algorithme DSA utilise une factorisation de grands nombres [34]. Il est utilisé par les autorités pour créer des signatures numériques. Cet algorithme ne s'étend pas au-delà du message proprement dit.

RSA est l'algorithme de cryptographie à clé publique le plus utilisé de nos jours. Créé en 1977 [35], cet algorithme tire son nom des initiales de ses trois inventeurs, à savoir Ron Rivest, Adi Shamir et Leonard Adleman. RSA dépend du chiffrement asymétrique. Outre la signature, cet algorithme chiffre le contenu du message.

DSA s'avère plus rapide que RSA pour la signature d'un document numérique. RSA, en revanche, convient mieux pour la signature et la vérification de documents électroniques et le chiffrement de messages.

Comme c'est généralement le cas dans le domaine de la cryptographie, l'algorithme RSA repose sur deux principes mathématiques, à savoir la factorisation de nombres premiers et de modules.

ECDSA est l'algorithme de signature numérique le plus récent. Il remplace progressivement l'algorithme RSA. L'avantage de ce nouvel algorithme est de pouvoir utiliser des clés bien plus petites pour le même niveau de sécurité et de nécessiter moins de puissance de calcul que RSA.

ECDSA est l'algorithme de signature numérique qui est utilisé pour assurer la signature numérique [36]. Ainsi nous allons aborder en détail la notion de courbe elliptique.

1.5.2 Cryptographie à courbe elliptique (ECC)

Blockchain utilise la cryptographie asymétrique où deux clés différentes (une clé publique et une clé privée) sont nécessaires [37]. Ces clés sont utilisées pour le cryptage, le décryptage et les signatures numériques. La plupart des plateformes Blockchains utilisent une courbe elliptique sur champ primaire pour la création de paires de clés et d'autres opérations comme la signature numérique. La cryptographie par courbe elliptique (ECC) présente une meilleure efficacité de mise en œuvre et une meilleure sécurité que d'autres schémas de cryptographie comme RSA et DSA. Elle est également plus adaptée aux dispositifs à faible puissance, à faible capacité de mémoire et de bande passante [38].

La sécurité des crypto-monnaies à courbe elliptique est basée sur la difficulté de leur problème logarithme discret. Cela signifie que, étant donné un point P sur une courbe elliptique, il est facile de multiplier P par un multiplicateur k pour obtenir un autre point Q. Cependant, il est très difficile (infaisable en termes de calcul avec la puissance de calcul actuelle) d'obtenir le multiplicateur en connaissant simplement les deux points P et Q. Par conséquent, en utilisant ECC, il est infaisable d'obtenir la clé privée de quelqu'un en connaissant simplement sa clé publique et le point générateur de la courbe.

1.5.2.1 Définition

Une courbe elliptique E sur le champ Fp est définie par l'équation [39] :

$E : y\ 2 = x\ 3 + ax + b\ mod\ p$ où : $a, b \in$ Fp : $4a\ 3 + 27b\ 2 \neq 0$.

Les paramètres a, b, p, G, n, h sont les paramètres du domaine global choisis pour une courbe elliptique particulière afin de déterminer les caractéristiques de la courbe. Le paramètre p est généralement un grand nombre premier servant de limite supérieure pour les coordonnées x et y. Le G est le point générateur qui est le point de base de la courbe qui sert à générer tous les autres points. Les n et h sont respectivement l'ordre de la courbe (déterminant le nombre de points sur la courbe) et le cofacteur de la courbe (#E(Fp)/n). Pour la sécurité, le n est généralement choisi comme une très grande valeur entière. Toutes les parties d'une même application ECC doivent utiliser les mêmes paramètres de courbe elliptique. Une courbe elliptique d'ordre n à n-1 points discrets à partir de 1 et incluant un point à l'infini, c'est-à-dire :

$\langle G \rangle = \{\infty, 1G, 2G, 3G..... (n - 1)\ G\}$

Créer une annexe qui renvoie aux détails qui évoque les ECDSA de la génération à la vérification de la signature numérique.

1.6 La problématique de la sécurité informatique

Dans le domaine de l'informatique, la sécurité consiste à garantir que les ressources à la fois matérielles ou logicielles d'une entreprise ou organisations soient utilisées dans le cadre prévu. En d'autres termes, elle couvre les ressources informatiques mises en œuvre pour réduire le vol et/ou la fuite d'information, la modification des données et les pannes de services. Par conséquent, elle

permet de garantir que les ressources soient utilisées dans le cadre dédié. Car il convient pour toute organisation de protéger son patrimoine qui est représenté essentiellement par son système d'information.

Ainsi on considère qu'un système informatique est sécurisé s'il incorpore des mécanismes et des services de sécurité permettant de faire face aux vulnérabilités, menaces, risques et attaques.

1.6.1 Vulnérabilités, Menaces, Risques et Attaques

On définit :

- **Une vulnérabilité**

Elle est considérée comme étant une faiblesse qui rend un système vulnérable aux attaques. Cependant il faut signaler que la plupart des attaques qui ont été menées contre la technologie Blockchain ne remettent pas en question la technologie en tant que telle [40]. La plupart des attaques qui ont été menées sont dues à des erreurs de codage. Ces failles peuvent être de plusieurs types : les failles physiques, les failles liées aux réseaux, les failles webs, les failles qui sont liées aux gens qui travaillent dans l'organisation ou entreprise et enfin les failles liées aux applications et systèmes ;

- **Une menace**

Elle est la probabilité qu'un événement nuisible, tel qu'une attaque, se produise. On distingue deux types de menaces à savoir la menace interne ou une menace externe. Certaines de ces menaces peuvent être accidentelles (incendies ou inondations) ou intentionnelles (en cas de volonté manifeste) de mettre en péril le système. Car les attaques peuvent provenir de l'intérieur comme de l'extérieur d'un réseau. Les menaces auxquelles les utilisateurs de la technologie Blockchain peuvent faire face, c'est le plus souvent le vol des données personnels et de leur bien (argent en volant ou en dérobant leur clé privé) en d'autre terme leur portefeuille, que les transactions lancées ne s'exécutent pas rapidement ou menace réseau et de minage [41] ;

- **Un risque**

Il dépend de la valeur de ce que nous tentons de protéger, ainsi il désigne la possibilité qu'un incident survienne après la concrétisation d'une menace. Par exemple, une menace accidentelle (inondation ou incendie) conduirait à une perte de matériel informatique. Les cybercriminels

tentent souvent d'exploiter la moindre faille dans le réseau Blockchain afin de commettre des attaques qui impliquent des détournements financiers (on parle de millions de dollars voire plus) sans être inquiétés. Dans la technologie Blockchain, les utilisateurs risquent d'être dépouillés de leurs biens [42]. D'où la nécessité de bien analyser les risques liés à un système. Ce qui conduirait à mettre en place une bonne politique de sécurité qui tiendrait compte des coûts également ;

- **Une attaque**

Elle est l'exploitation délibérée d'une faiblesse trouvée dans un système informatique comme cible spécifique ou simplement comme cible non intentionnelle. Ainsi différentes raisons peuvent expliquer le choix des cibles des cybercriminels. Pour ce faire, les cybercriminels cherchent et identifient en permanence les systèmes présentant des vulnérabilités flagrantes à l'image des systèmes non protégés. Ainsi des attaques ont été menées et nous allons revenir sur ces attaques dans la section suivante.

1.7 Les services de la sécurité informatique

Pour s'assurer du bon fonctionnement d'un système informatique, on doit mettre en place des services de qualité, à savoir :

- **L'authentification :** elle consiste à l'identification de l'utilisateur pour la gestion des accès à des espaces de travail pertinents et de maintenir la confiance dans les relations. De cette manière, dans le cadre de la Blockchain, elle permet aux utilisateurs d'effectuer entre autres des transactions financières ;
- **L'intégrité :** c'est une propriété de la politique de sécurité. Elle fait généralement référence à la prévention de tous types de destruction ou de modification d'informations par des parties non autorisées. La menace d'intégrité comprend généralement une menace liée aux finances. Par exemple la modification des données financières, le vol d'argent, le réacheminement du dépôt ou le détournement ;

  Il représente l'exactitude, la cohérence et la fiabilité des données pendant tout leur cycle de vie [43] . Un autre terme utilisé pour l'intégrité est la qualité. En effet, les données font l'objet de nombreuses opérations telles que la capture, le stockage, la récupération, la mise

à jour et le transfert. Cependant, elles ne doivent, à aucun moment, être altérées par des entités non autorisées. Les méthodes utilisées pour assurer l'intégrité des données comprennent le calcul de hash (nous allons y revenir plus tard dans la suite du document), les contrôles de validité des données et les contrôles d'accès.

- **La confidentialité :** c'est une autre propriété de la sécurité qui fait référence à la protection des informations et des systèmes contre les parties non autorisées. La menace de confidentialité peut généralement cibler les bases de données, les serveurs d'applications, les administrateurs systèmes et peut être considéré comme un « vol de données » ;
- **La disponibilité :** c'est un service de sécurité qui équivaut à garantir l'accès des systèmes d'information ou des actifs à une partie ou une entité autorisée de manière fiable et rapide. La menace de disponibilité comprend généralement le déni de service ou la destruction physique, et peut être considérée comme un « refus d'accès aux données ». Ainsi certains dysfonctionnements et attaques empêchent l'accès aux systèmes et services d'informations ;
- **La non-répudiation :** Dans le jargon juridique, la répudiation est synonyme de renonciation. La non-répudiation est le fait de s'assurer que l'expéditeur d'un message ou document ne peut nier le fait qu'il l'a envoyé et que le destinataire ne peut nier le fait qu'il l'a reçu.

Malgré la mise en place de ces services de qualité dans la technologie Blockchain. Nous constatons qu'elle comporte beaucoup de risques et des limites qui sont développés dans ce qui suit.

1.8 Conclusion

En définitive, nous avons passé en revue l'ensemble des notions de cryptographie qui sous-tendent la technologie blockchain. D'abord les mécanismes de sécurité à travers la cryptographie symétrique et asymétrique en passant les fonctions de hachage et la signature numérique, tout en mettant l'accent sur les ECCs qui constituent des éléments indispensables dans la technologie blockchain. Ensuite nous avons parlé des problématiques de la sécurité informatique que pourrait rencontrer la technologie Blockchain. Pour enfin terminer par évoquer les services de sécurité informatique.

Toutes ces notions sont essentielles pour la bonne compréhension de la technologie Blockchain que nous allons aborder dans le chapitre suivant. C'est pourquoi nous émettons quelques interrogations à ce sujet :

Quels sont les impacts de la sécurité informatique pour la technologie Blockchain ?

Est-ce qu'ils permettent de rendre le système infaillible ?

Nous allons essayer d'apporter des réponses à ces questions dans le chapitre suivant.

Chapitre II : état de l'art de la technologie Blockchain

Etat de l’art de la technologie Blockchain

La technologie Blockchain est à la fois émergente, inspirante et capable de perturber de nombreuses industries et notre mode de vie. Outre les crypto-monnaies, la blockchain peut aider de nombreuses industries à améliorer les inefficacités et à surmonter de nombreux goulots d'étranglement. La technologie a été adoptée dans plusieurs pays (à l’image de la Géorgie, de l'Estonie et de la Russie) et dans des entreprises telles qu'IBM et Microsoft.

La plupart des projets d'entreprise sur la Blockchain sont actuellement en phase de production, même si d'autres ont été déployés. Parmi ces entreprises, nous pouvons citer Cisco (spécialisée dans le matériel réseau), Facebook (qui fournit principalement des sites de réseaux sociaux à mis en place sa monnaie numérique Libra).

Pour mieux comprendre cette nouvelle technologie, il est nécessaire de l’examiner dans le but de comprendre les termes basiques et son fonctionnement de manière très détaillée.

Après nous allons montrer les bases de la confiance en la technologie Blockchain. Sans oublier les risques qu’elle engendre.

Ainsi nous terminerons par présenter les attaques contre la technologie Blockchain et les différents domaines d'application de la technologie Blockchain.

2 Qu'est-ce que la blockchain

2.1 Historique

La Blockchain est la technologie sous-jacente à la crypto-monnaie, issue de la vérification des horodatages numériques à la fin des années 1980 et au début des années 1990 [44].

En 1990, Haber et Stornetta ont publié un article intitulé « How to Timestamp a digital Document » [45]. Dans cet article, ils proposent de créer une chaîne de hachage liant les horodatages publiés entre eux afin que le document ne soit pas obsolète ou reculé. A la fin de l'année 1992, Haber, Stornetta et Dave Bayer ont ajouté le concept des arbres de Merkle à cette conception [46]. Les arbres Merkle aident à améliorer l'efficacité du système en collectant plusieurs documents horodatés dans une blockchain cryptée et sécurisée. Chaque enregistrement précédent aide dans les enregistrements récents à comprendre l'histoire de toute la chaîne. Puis, Dai Wei, l'un des célèbres chercheurs, avait introduit le concept de b-money [47] pour créer de la monnaie en résolvant des problèmes informatiques et en décentralisant le consensus. Cependant, la proposition manque de détails de mise en œuvre.

En 2005, un concept appelé « Reusable Proof of Work » (RPoW) a été introduit par le militant du chiffrement Hal Finney [48]. Ce concept combine l'idée de b-money avec le difficile problème de calcul de Hashcash qu'Adam Back a proposé pour créer une crypto-monnaie. RPoW enregistre la propriété des jetons sur des serveurs de confiance. Ces serveurs permettent aux utilisateurs de vérifier l'exactitude et l'exhaustivité des utilisateurs, résolvant ainsi le problème de la double dépense.

En 2008, un mystérieux livre intitulé « Bitcoin : A peer to peer Electronic Cash system » [49] écrit par un visionnaire Satoshi Nakamoto a donné naissance au concept de blockchain. Dans cet article, Satoshi Nakamoto combine cryptographie, informatique et théorie des jeux pour décrire la monnaie numérique bitcoin. Cela aide les participants à effectuer des transactions d'un compte à un autre sans l'aide d'intermédiaires tels que les agences centrales ou des banques. La chronologie suivante illustre brièvement l'émergence [46]de la technologie blockchain.

Année	Émergence de la blockchain.
1990	Stuart Haber et Stornetta ont introduit l'horodatage d'un document numérique afin qu'il ne puisse pas être falsifié.
1992	Le concept des arbres de Merkle a été proposé pour rassembler plusieurs documents en un seul bloc.
2000	La théorie et l'idée des chaînes sécurisées cryptographiques ont été proposées par Stefen Konst.
2005	Hal Finney a présenté la « preuve de travail réutilisable » (RPoW) qui aide les utilisateurs à résoudre le problème de double dépense dans la création de crypto-monnaies.
2008	Satoshi Nakamoto a proposé Bitcoin, une monnaie numérique qui utilise Blockchain comme concept sous-jacent.

Tableau 3: Chronologie de l'émergence de la blockchain

2.2 Définition

La technologie Blockchain n’est pas une technologie unique. Comme rappelé dans [50] , elle comprend la cryptographie, les mathématiques, les algorithmes et les modèles commerciaux,

combinés avec des réseaux pairs-à-pairs. Elle utilise des algorithmes de consensus distribués pour résoudre les problèmes de synchronisation de base, infrastructure multi-domaines intégrée.

En d'autres termes [51], la Blockchain est un grand livre décentralisé qui est géré entre pairs sans l'intervention d'un organisme central, assurant ainsi la transparence et la sécurité du système. Pour visualiser comment cela fonctionne, nous pouvons voir la blockchain comme un registre ou un grand livre ou tout est enregistré dans les moindres détails. Ainsi la traçabilité de tous les échanges peut être maintenue.

La technologie Blockchain présente les caractéristiques suivantes :

- **Décentralisée**

La décentralisation est une caractéristique de base de la Blockchain. Cela signifie que la technologie Blockchain n'a pas à s'appuyer sur un nœud ou un élément centralisé. Les données peuvent être enregistrées, stockées et mises à jour de manière distribuée.

- **Transparence**

L'enregistrement des données par le système Blockchain est transparent pour chaque nœud, il est également transparent sur la mise à jour des données. C'est pourquoi la Blockchain peut être fiable.

- **Source ouverte**

La plupart des systèmes Blockchain sont ouverts à tous, les enregistrements peuvent être vérifiés publiquement. Les gens peuvent aussi utiliser la technologie Blockchain pour créer n'importe quelle application.

- **Autonomie**

En raison de la base du consensus, chaque nœud sur le système de Blockchain peut transférer ou mettre à jour les données en toute sécurité. L'idée est de ne pas faire confiance à une seule personne pour l'ensemble du système, et personne ne peut intervenir.

- **Immuable**

Tout enregistrement sera réservé pour toujours et ne peut pas être modifié. A moins que quelqu'un puisse prendre le contrôle de plus de 51% des nœuds en même temps.

- **Anonymat**

La technologie Blockchain a résolu le problème de confiance entre les nœuds, le transfert de données ou même la donnée ou même les transactions peuvent être anonymes. Il suffit de connaître l'adresse Blockchain de la personne.

Ainsi nous allons passer en revue les différentes notions de base de la technologie Blockchain. Mais avant cela, nous allons parler de l'architecture en couches de la technologie Blockchain.

2.3 Architecture en couche de la technologie blockchain

Comme vous pouvez le constater, l'architecture de la technologie Blockchain comprend cinq couches, illustrées à la Figure *6*, utilisée dans la plupart des applications existantes.

- **La couche réseau** : elle permet d'établir la communication entre les entités participantes de réseau Blockchain pair-à-pair. Elle se comporte comme une couche de commérage [52].

- **La couche de consensus** : Elle garantit le fait que les blocs générer par les différents nœuds soient ajouter dans le bon ordre dans la chaine. A titre d'exemple nous pouvons cité : le PoW et le PoS (*voir la section 2.6*). La couche de consensus a pour fonction de faire accepter par tous les nœuds du système les informations fournies et stockées par les nœuds du réseau Blockchain.

- **La couche de données** : Elle a pour fonction de définir le modèle de données et le stockage physique. La technologie étant une forme de base de données transparente et autorégulatrice à code source ouvert et répliqué.

- **La couche d'exécution** : Elle joue le rôle d'exécution des Blockchain qui intègre les smarts contracts. Elle est apparue avec la Blockchain Ethereum avec sa machine virtuelle (EVM) [53] [54] [55] qui facilite l' exécution des smarts contracts.

- **La couche applicative** : Elle comporte de nombreuse application qui facilite la communication entre les utilisateurs et la Blockchain. Le réseau Blockchain constitue une structure dorsale pour ces applications. Les enregistrements de la chaîne de blocs sont irréversibles et ouverts aux participants, ce qui garantit qu'ils ne peuvent jamais être modifiés une fois qu'un enregistrement est incorporé, et cet aspect favorise **l'intégrité des données** [56].

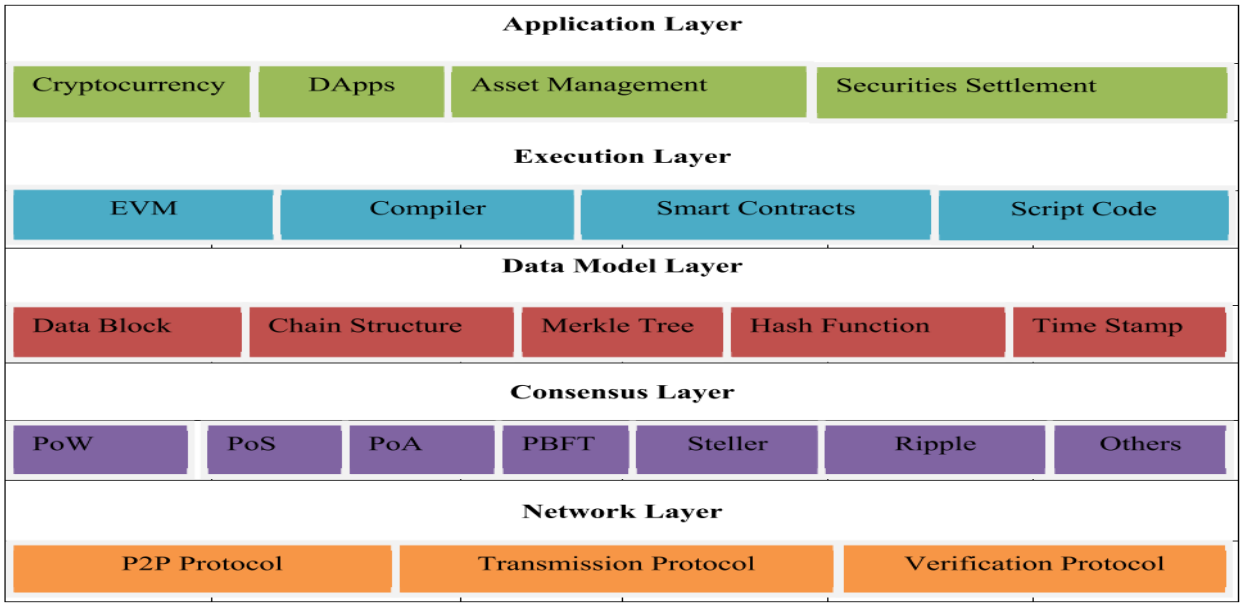


Figure 6 : Architecture en couche

2.4 Les notions de base

2.4.1 Transaction

Nous expliquons le concept de la chaîne de blocs en expliquant comment le Bitcoin fonctionne puisqu'il est intrinsèquement lié au Bitcoin (le Bitcoin est la mère de toutes les Blockchains). Cependant, la technologie Blockchain est applicable à toute transaction d'actif numérique échangée en ligne. Elle permet de manière générale d'assurer ses trois fonctions, à savoir :

- Valider les entrées ;
- Sauvegarde des entrées ;
- Préserver le dossier historique.

Le commerce sur internet est associé aux institutions financières, et ces dernières agissent en tant que tiers de confiance pour traiter et arbitrer toutes les transactions électroniques. Le rôle d'un tiers de confiance est de vérifier, sauvegarder et préserver les transactions. Un certain pourcentage de fraude est inévitable dans les transactions en ligne. Cela nécessite une médiation par des transactions financières, entraînant des coûts élevés.

Bitcoin utilise la preuve cryptographique plutôt qu'un mécanisme de confiance tierce afin que deux parties contractantes puissent effectuer des transactions en ligne via internet.

Chaque transaction est protégée par une signature numérique, transmise à la clé publique du destinataire et signée numériquement en utilisant la clé privée de l'expéditeur.

L'entité recevant la monnaie numérique vérifie ensuite la signature numérique. Cela signifie que l'entité recevant vérifie la propriété de la clé privée correspondante et l'utilisation de la clé publique de l'expéditeur dans la transaction correspondante. Chaque transaction est diffusée à chaque nœud du réseau bitcoin, puis enregistrée dans le grand livre public. Avant d'enregistrer la transaction, le nœud de validation doit s'assurer de deux choses :

- L'expéditeur est propriétaire de la crypto-occurrence, grâce à la vérification de la signature numérique de la transaction.
- L'expéditeur a suffisamment de crypto-monnaies sur son compte, grâce à la vérification de chaque transaction par rapport au compte de l'expéditeur. Cette opération de vérification est faite à l'aide de la clé publique du dépensier qui est enregistrée dans le grand livre. Cela garantit qu'il y a un solde suffisant sur son compte avant de finaliser la transaction (voir section 2.4.2).

Cependant, il existe un problème de régulation de ces transactions valide qui est diffusé à tous les autres nœuds du réseau bitcoin pair-à-pair. Les transactions ne sont pas effectuées dans l'ordre dans lequel elles ont été générées, il est donc nécessaire d'établir un système pour s'assurer que la double dépense de crypto-monnaies ne se produise pas. En considérant que les transactions sont transmises nœuds par nœud à travers le réseau bitcoin, il n'y a aucune garantie que l'ordre dans lequel elles sont reçues à un nœud soit du même ordre que celui dans lequel ces transactions ont été générées. Ce qui précède signifie qu'il est nécessaire de développer un mécanisme afin que l'ensemble du

réseau bitcoin puisse se mettre d'accord sur l'ordre des transactions, ce qui est une tâche ardue dans un système distribué.

Bitcoin a résolu ce problème par un mécanisme qui est maintenant populaire et connu sous le nom de technologie Blockchain. Le système Bitcoin ordonne les transactions en les plaçant dans des groupes appelés blocs, puis en reliant ces blocs, ce chaînage confère son nom à la Blockchain. Les transactions dans un bloc sont considérées comme ayant été produites au même moment. Ces blocs sont liés les uns aux autres (comme une chaîne) dans un ordre linéaire et chronologique. Chaque bloc contenant le hachage du bloc précédent.

Il reste encore un problème : n'importe quel nœud du réseau peut collecter des transactions non confirmées et créer un bloc. Ce bloc sera ensuite diffusé au reste du réseau en tant que suggestion sur le bloc à être le prochain dans la Blockchain.

2.4.2 Comment le réseau décide-t-il qu'un bloc devrait être le suivant dans la blockchain ?

Il peut y avoir plusieurs blocs créés par différents nœuds en même temps. On ne peut pas se fier à la commande car les blocs peuvent arriver à des ordres différents et à différents points du réseau.

Bitcoin résout ce problème en introduisant **un casse-tête mathématique**. **Ainsi un bloc sera accepté dans la Blockchain à condition qu'il contienne une réponse à un problème mathématique très particulier**. Ceci est également connu sous le nom de « Proof of Work **»** : un nœud générant un bloc doit prouver qu'il a mis assez de ressources informatiques pour résoudre une énigme mathématique. Par exemple, on peut demander à un nœud de trouver **un Nonce qui, lorsqu'il est haché avec les transactions et les hachages de blocs précédents produit un hachage avec un certain nombre de zéros de tête**.

L'effort moyen requis est exponentiel au nombre de zéros bits requis mais le processus de vérification est très simple et peut être fait en exécutant un seul hachage [57].

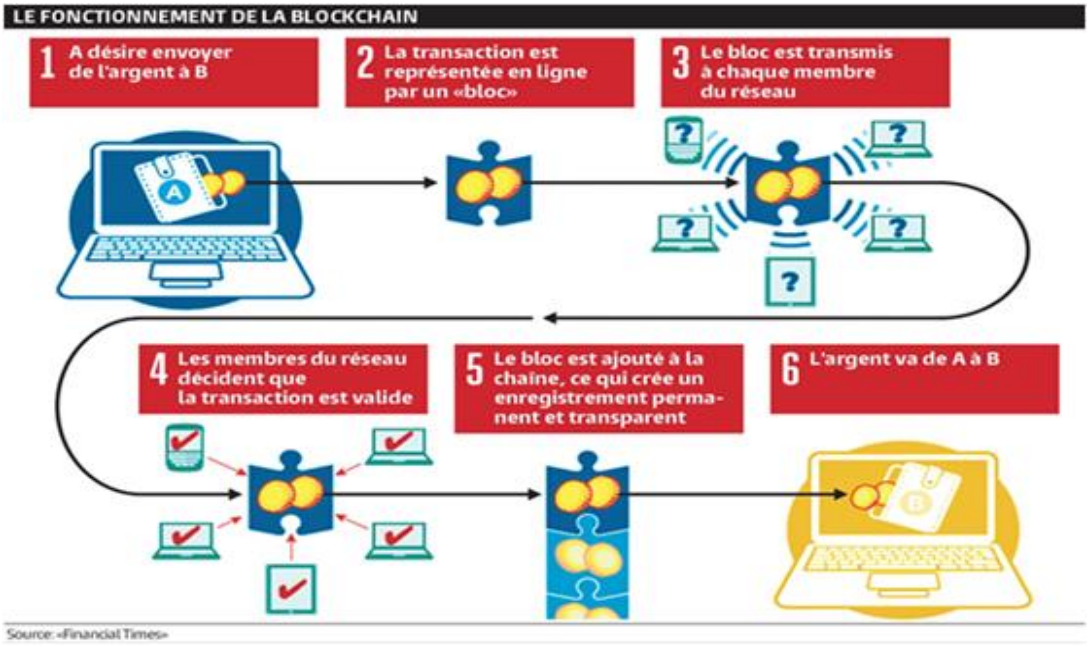


Figure 7 : Le mode de fonctionnement d'une Blockchain *[58]*

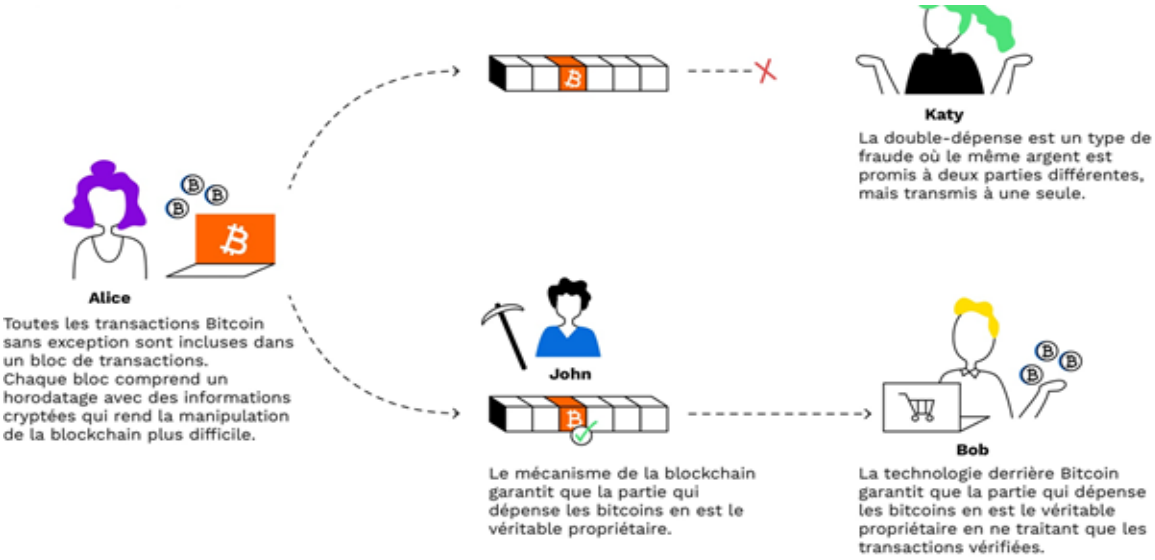


Figure 8 : Résolution du problème de la double dépense *[59]*

2.4.3 Processus de vérification et de validation d'une transaction

Le principal défi d'un réseau décentralisé est de parvenir à un accord sur la validité des transactions. Valider une transaction est un processus à plusieurs niveaux : il faut identifier l'émetteur comme légitime (rappelons que tout utilisateur de bitcoin interagit avec le réseau de manière anonyme), vérifié qu'il possède suffisamment de fonds pour effectuer la transaction et ordonner

chronologiquement les transactions. Chaque nœud peut vérifier ces conditions de manière indépendante assez facilement mais, au niveau du réseau, il est beaucoup plus compliqué de trouver un consensus. En effet, n'importe quelle personne possédant un ordinateur peut se connecter de façon anonyme et participer au maintien du registre, ce qui implique qu'un nœud ne peut a priori pas faire confiance aux autres.

Permettre à un système décentralisé de s'entendre sans que les participants aient besoin de se faire confiance est, en résumé, ce que la technologie du Blockchain rend possible.

D'où la nécessité d'identifier l'émetteur de la transaction comme légitime.

- Identification de l'émetteur de la transaction

    - La paire de clé : clé privée/ clé public

Théoriquement, lorsqu'un nœud reçoit une transaction transférant une certaine quantité d'argent d'Alice à Bob, il n'a aucun moyen de savoir si Alice est l'émettrice de cette transaction.

Pour régler ce problème, Bitcoin utilise un système de cryptographie commun basé sur une combinaison clé privée / clé publique. La clé publique, ou adresse, est visible pour tous (pour l'instant nous ne faisons pas trop la différence entre clé publique et adresse). Concrètement, quand Alice fait une transaction pour Bob, c'est à son adresse qu'elle envoie l'argent.

Cette adresse est en relation directe avec la clé privée de Bob, qui est en quelque sorte le mot de passe associé à cette clé. En effet, l'adresse est générée à partir de la clé privée grâce à des fonctions mathématiques complexes qui permettent de se souvenir qu'elles **fonctionnent dans un sens en d'autres termes irréversibles** [60]. Par conséquent, en appliquant ces fonctions à une clé privée (P), nous obtiendrons toujours la même adresse (A), mais il n'est pas possible, sachant (A), de trouver (P). Le possesseur de (P) peut donc aisément prouver qu'il détient les fonds entreposés à (A).

Est-ce que le problème est réglé ? Non, pas encore. Il est important de comprendre que ce système fonctionne de façon décentralisée. Cela implique que vous ne pouvez pas simplement donner votre clé privée au réseau pour prouver que vous possédez les fonds, comme vous donneriez votre mot de passe à une banque en ligne. En effet, un nœud qui recevra votre clé privée serait en mesure de

la réutiliser pour dépenser votre argent (de même que la banque en ligne pourrait en théorie dépenser vos fonds, puisqu'elle connaît votre mot de passe). Il faut donc trouver une manière de prouver au réseau que vous avez la clé privée, sans fournir la clé elle-même. Cette pièce à conviction est la signature numérique.

L'idée est simple : plutôt que de fournir la clé privée au réseau, nous fournissons une signature numérique générée à partir de la clé privée et de la transaction à l'étude. La signature numérique est la preuve que l'émetteur avait la clé privée associée à l'adresse d'émission lors de la transaction. Cette signature est transmise avec la transaction au réseau, ce qui permet à chaque nœud de valider ce dernier sans avoir accès à la clé privée de l'émetteur.

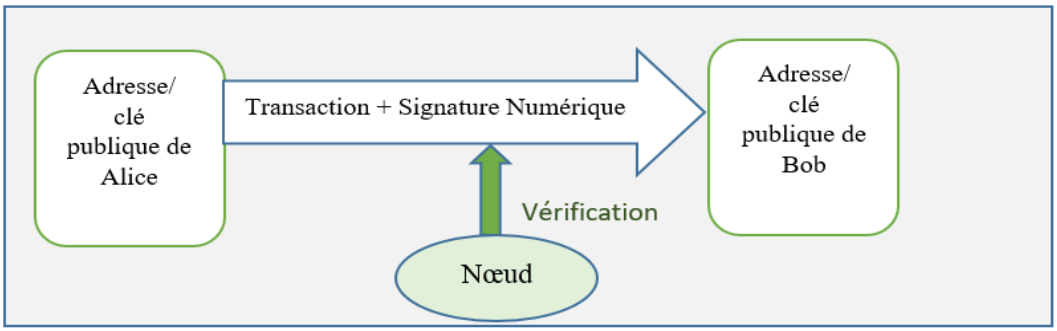


Figure 9 : Vérification et Validation d'une transaction 1

C'est une bonne idée, mais on ne vient pas régler le problème. Est-ce que le nœud ne peut pas réutiliser une signature pour valider une autre opération à partir de la même adresse ?

Non. Cette fois, c'est impossible. En effet, une signature digitale est différente de chaque transaction émise, de telle sorte qu'il est impossible de prendre la signature digitale d'une transaction pour tenter de valider une autre transaction. Pour y voir un peu plus clair, intéressons-nous de plus prêt à l'algorithme de vérification d'une transaction.

En résumé, une signature numérique est une fonction F qui prend comme paramètres la clé privée de l'expéditeur (p) et le message envoyé (m). Pour vérifier si la signature est valide, chaque nœud applique une autre fonction V qui renvoie vrai ou faux avec l'adresse publique de l'expéditeur (a), le message (m) et la signature numérique (s) comme paramètres. Notez que la fonction F est à sens unique, il est donc impossible de deviner la clé privée à partir de la signature. Cela implique également une interdépendance entre la signature et le message : F donnera toujours la même

signature pour un message et une clé privée donnés, mais le moindre changement dans l'un des deux paramètres aura pour effet de changer radicalement la signature. Par conséquent, les signatures numériques ne peuvent jamais être utilisées. Cela empêche également les nœuds relayant les transactions de les modifier en cours de route.

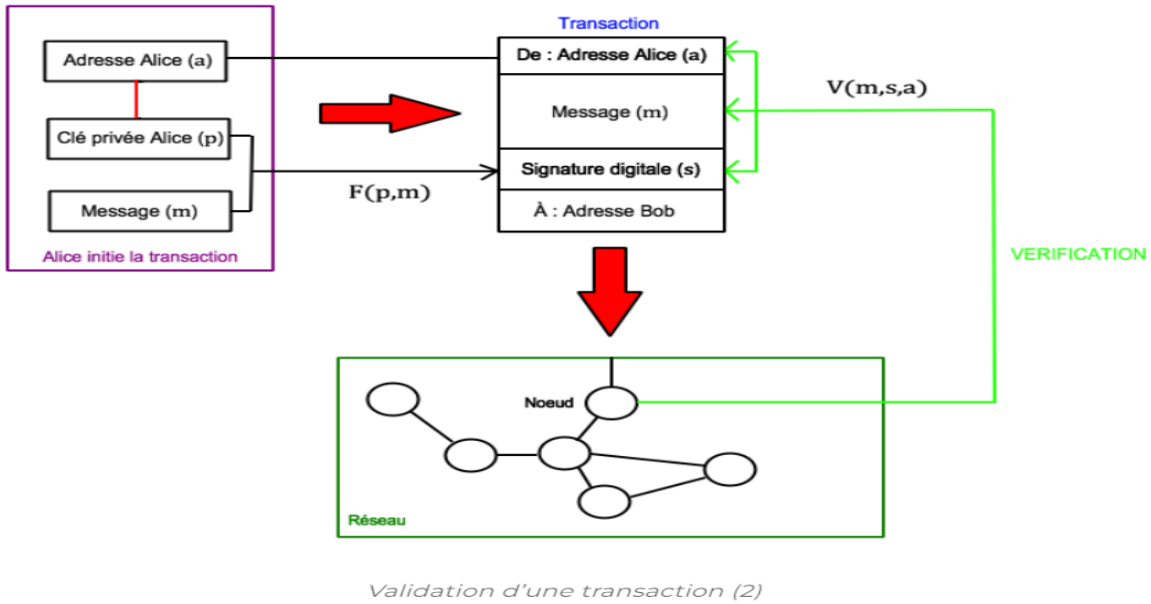


Figure 10 : vérification et validation d'une transaction 2

NB : sur le schéma de la transaction, l'adresse du destinataire apparaît distincte du message. En réalité, on peut considérer qu'elle est incluse dans m, de telle sorte qu'un nœud ne puisse pas la modifier sans invalider la signature.

Rigoureusement parlant, la clé publique et l'adresse bitcoin ne signifient pas la même chose. Cependant elles ont la même fonction. En effet, comme décrit dans [61], une adresse bitcoin est la version légère de la clé publique qui est adaptée à l'utilisation. Pour obtenir une adresse, il suffit d'utiliser quelques conversions mathématiques à la clé publique, et le tour est joué.

Par ailleurs, la paire clé privé/clé publique est créée en dehors du réseau Blockchain. En d'autres termes, le réseau ne parcourt pas la liste des adresses déjà utilisées afin de proposer une valide. Car dans le mécanisme de génération d'une adresse, nous avons vu tantôt que cela était intrinsèquement lié à la clé publique. Qui à son tour a été obtenue à partir de la clé privée. Par conséquent, si le

réseau veut fournir une adresse valide, il doit générer une clé privée. Une telle tâche n'est pas facilitée par le système décentralisé voire impossible.

Par contre une autre question subsiste, est-il possible que deux personnes aient la même paire clé privée/ clé publique ?

La réponse est qu'en théorie Oui, de ce fait deux personnes distinctes auront la possibilité d'avoir accès au fond stocké à l'adresse concernée. Cependant je vous rassure que cette probabilité est inexistante (voir quasi nulle).

Preuve : soit 2^{160} le nombre d'adresse totale bitcoin possible. En comparaison, il y a environ 2^{63} grains de sable sur terre. Si chacun de ces grains contenait une terre avec autant de grain de sable. Alors le nombre total de grains de sable serait de 2^{126}, une valeur qui est encore très loin de nos 2^{160}. Ce qui prouve qu'il est mathématiquement impossible (pour l'instant), qu'une personne génère une clé privée qui par la suite donnerait une adresse déjà utilisée.

À cela s'ajoute le fait que, nos ordinateurs (avec leurs capacités de calcul actuel) parviennent à générer des milliers de clés privées. Pour ensuite prouver que si l'adresse équivalente contient de l'argent, qu'il va ensuite transférer à son adresse. Cette attaque est quasi impossible, car il lui faudrait attendre des décennies pour obtenir un résultat.

Exemple : supposons que l'attaquant ait une machine avec une puissance de calcul très élevée, et capable de générer environ 2^{30} clés privées par seconde. Ce qui implique qu'il lui faudrait en moyenne2^{130} secondes pour trouver la bonne clé. Même si on suppose qu'il possède plusieurs de ces ordinateurs (1 milliard de milliards), il lui faudrait alors 2^{70} secondes ou 2^{45} années. Sachant que l'univers est âgé de 2^{24} années, alors là nous lui souhaitons vraiment bonne chance.

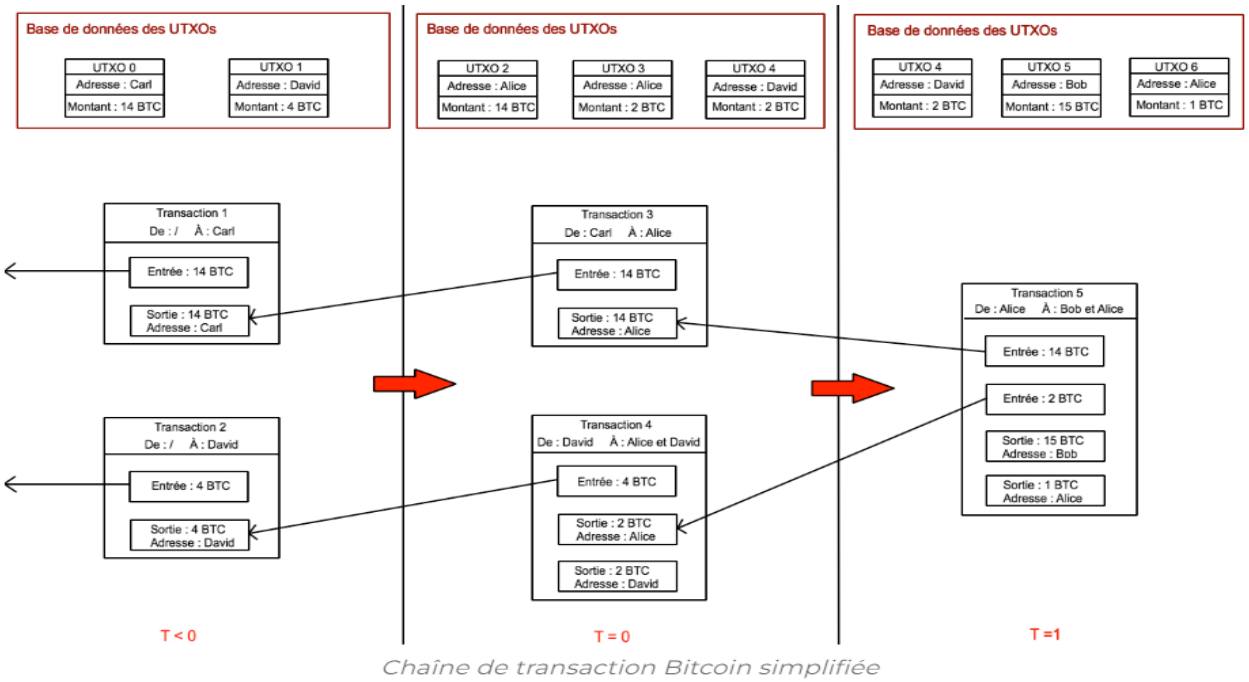
- Comment puis-je m'assurer que l'expéditeur à suffisamment d'argent dans son compte ?

Contrairement aux banques traditionnelles, le registre tenu par le réseau de nœuds ne contient pas les soldes associés à chaque adresse. Afin de calculer le solde d'un compte (note : chaque compte définit une paire de clés privée/publique), le nœud associe la « sortie de transaction non dépensée » (sortie de transaction non dépensée ou UTXO) associée à l'adresse du compte considéré. Derrière ce nom sauvage se cache un fait relativement simple : pour prouver sa validité, toute transaction

référence une ou plusieurs transactions passées comme entrée dont la somme des sorties est supérieure ou égale à son nombre. Ce mécanisme de référence conduit à la formation de chaînes de transactions, qui ont une propriété de base : une transaction peut référencer plusieurs UTXOs en tant qu'entrées et sorties, mais un UTXOs particulier ne peut être utilisé qu'une seule fois en tant qu'entrée unique.

Cela permet d'empêcher que la même somme d'argent ne soit dépensée deux fois (voir Figure *8*). Les UTXOs sont stockés dans une base de données et chaque nœud en possède une copie. Pour chaque transaction vérifiée, la base de données est mise à jour.

Pour une meilleure compréhension, considérons l'exemple suivant. Supposons qu'Alice veuille envoyer 15 BTC à Bob. Pour ce faire, il doit saisir une ou plusieurs transactions qui lui ont été envoyées lors de l'initiation d'une transaction, dont les sorties n'ont pas été dépensées et dont la valeur totale est supérieure ou égale à 15 BTC. Le schéma suivant illustre (voir Figure *11*), de manière très simplifiée, le fonctionnement de la chaîne transactionnelle qui y conduit. Nous supposons qu'à T < 0, la base de données UTXOs ne contient que deux UTXOs, un valant 14 BTC attribué à l'adresse de Carl et un valant 4 BTC attribué à l'adresse de David. A T=0, Carl et David exécutent la transaction destinée à Alice, puis à T=1 Alice exécute la transaction destinée à Bob.



Chaîne de transaction Bitcoin simplifiée

Figure 11 : Chaîne de transaction simplifiée dans Bitcoin

Comme indiqué, toute transaction doit indiquer une ou plusieurs entrées, chacune faisant référence à une sortie de transaction inutilisée (c'est-à-dire existant dans la base de données UTXO), et dont l'adresse correspond à l'adresse de la transaction. Afin de pouvoir utiliser cet UTXO, l'expéditeur doit fournir une signature numérique prouvant qu'il possède bien la clé privée associée à l'adresse UTXO.

Notez également que pour chaque transaction, la somme des entrées est égale à la somme des sorties. Par exemple, pour la transaction 5, Alice veut envoyer 15 BTC à Bob. Par conséquent, il doit se référer aux UTXOs dont la somme est supérieure ou égale à 15 BTC. Il est important de comprendre que les UTXOs sont indivisibles. Alice doit donc dépenser ses 2 UTXO, dont la valeur totale est de 16 BTC. Pour restaurer le changement de 1 BTC, Alice "crée" une sortie de 1 BTC attribué à son adresse.

Si la somme des entrées n'est pas égale à la somme des sorties, la différence (exportations totales - entrées totales) est considérée comme un frais de transaction et peut être distribuée aux mineurs de la transaction (nous verrons ce que cela signifie plus tard) bien entendu, la chaîne de transaction Bitcoin est en pratique bien plus importante que celle présentée ci-dessus. Lorsqu'un nœud se connecte pour la première fois au réseau, il doit créer sa base de données UTXOs. Pour ce faire, il vérifie toutes les transactions qui ont déjà été effectuées et enregistre toutes les sorties de transaction inutilisées dans la base de données. Ce processus est long et peut prendre plusieurs heures, mais il ne doit être fait qu'une seule fois. Après cela, un nœud synchronisera sa propre base de données en temps réel.

Nous pouvons maintenant exprimer le mécanisme par lequel les nœuds valident les transactions.

Pour chaque entrée dans la transaction :

- Renvoie une erreur si l'UTXO référencé n'est pas dans la base de données UTXO,
- Renvoie une erreur si la signature fournie ne correspond pas à la signature du propriétaire de l'UTXO (c'est-à-dire une adresse différente),
- Renvoie une erreur si la somme du nombre d'UTXO référencés dans l'entrée n'est pas égale à l'UTXO sorti,

- Mettre à jour la base de données UTXO.

Très bien, vous savez maintenant comment vérifier les transactions. Nous pouvons désormais explorer la Blockchain en toute confiance.

Ainsi nous allons voir en détail, comment les blocs sont formés et comment ils sont reliés les uns aux autres.

2.4.4 Les blocs

La Blockchain est une séquence de blocs qui contient une liste complète d'enregistrements de transactions. La Figure *12* illustre un exemple de Blockchain. Chaque bloc pointe vers le bloc précédent via une référence, qui est essentiellement la valeur de hachage du bloc précédent appelé bloc parent. Il est noté que la valeur de hachage du bloc oncle (enfant du bloc ancêtre) sera également stockée dans la Blockchain. Le premier bloc de la Blockchain s'appelle bloc de genèse, il n'a pas de bloc parent [62].

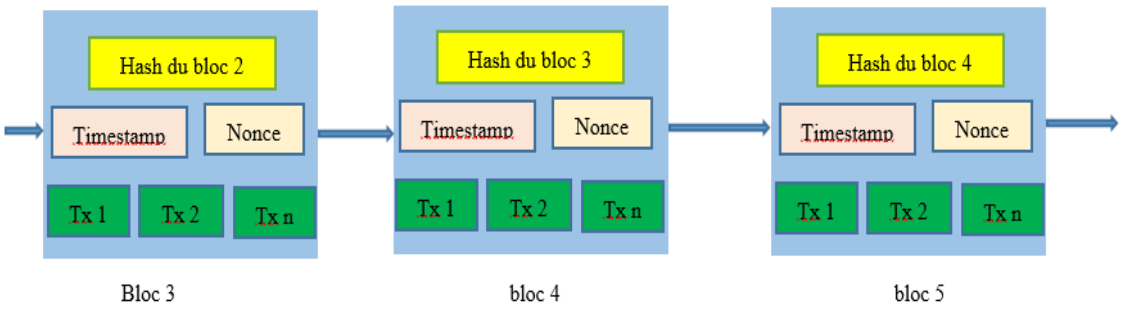


Figure 12 : Un Exemple de blockchain qui consiste en une séquence continue de blocs

Un bloc se compose de l'en-tête et du corps du bloc, comme le montre la Figure *13*. En particulier, l'en-tête de bloc comprend :

- **Version du bloc** : indique l'ensemble des règles de validation du bloc à suivre,
- **Le hachage du bloc parent** : une valeur de hachage de 256 bits qui pointe vers le bloc précédent,
- **Le hachage de la racine de l'arbre de Merkle** : la valeur de hachage de toutes les transactions du bloc,
- **Horodatage** : horodatage actuel en secondes depuis 1970-01-01T00:00 UTC,

- **nBits** : cible de hachage actuelle dans un format compact,
- **Nonce** : un champ de 4 octets, qui commence généralement par 0 et augmente à chaque calcul de hachage.

Le corps principal du bloc est composé de compteurs de transactions et de transactions. Le nombre maximum de transactions qu'un bloc peut contenir dépend de la taille du bloc et de celle de chaque transaction. La Blockchain utilise un mécanisme de cryptage asymétrique pour vérifier l'identité des transactions. Elle utilise des signatures numériques basées sur un chiffrement asymétrique dans un environnement non fiable.

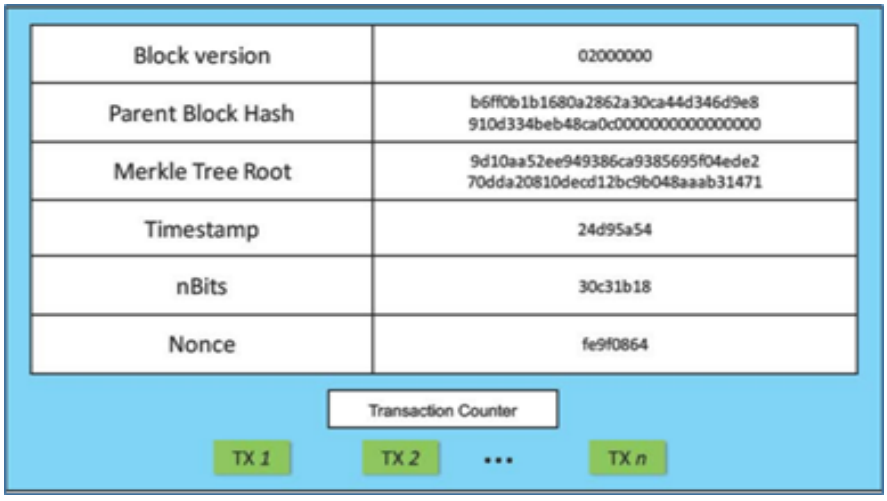


Figure 13: Structure d'un bloc

Comme illustré sur la Figure *12*, la Blockchain est une chaîne unique de flux de liens. Par contre si chaque mineur ajoute des blocs à la chaine, la chaine aura de nombreuses branches.
Ce qui implique que le système ne sera pas dans un état cohérent. Pour pallier cette problématique, les mineurs s'affrontent pour trouver le nouveau bloc et recevoir une récompense. **En fait, l'exploitation minière est une opération qui permet de trouver des blocs valides et de tirer profit de la résolution de problèmes mathématiques complexes**.
Une fois que les mineurs ont résolu le problème, des annonces sont diffusées sur le réseau et des blocs sont diffusés sur le réseau. Puis un autre mineur valide le nouveau bloc. Ils parviennent à un consensus pour ajouter un nouveau bloc à la chaîne. Ce nouveau bloc est ajouté à leur copie locale de la Blockchain.

L'algorithme de consensus est appelé protocole de preuve de travail (Proof-of Work), puisqu'il nécessite beaucoup de calculs de la part du mineur et consiste à trouver la valeur du champ **Nonce** de 32 bits à renseigner dans l'entête du bloc pour que le hachage de l'entête du bloc aboutisse à un résultat inférieur à une certaine valeur. **Plus cette valeur est petite, plus le problème est difficile à résoudre**. Pour conserver une même complexité calculatoire au fil du temps, il est intéressant que la Blockchain ajuste le niveau de difficulté. C'est le cas pour le projet Bitcoin qui s'appuie sur une moyenne de minage d'un bloc de 10 minutes. Ainsi, tous les 2016 [63] blocs qui correspondent à une période théorique de deux semaines, une moyenne est calculée ; si le temps moyen est trop court, la difficulté est alors revue à la hausse ; s'il est trop long, la difficulté est revue à la baisse. La nature probabiliste de cette sélection donne également lieu à la création de divergence (ou « forks ») dans la chaîne de bloc, qui doivent être résolues de façon univoque pour qu'un consensus stable soit rapidement achevé.

2.4.5 Nœuds

D'une manière générale, un nœud Bitcoin désigne tout ordinateur connecté au réseau bitcoin. En des termes simples, un ordinateur connecté au réseau bitcoin constitue un nœud. Mais en fait, ce n'est pas si simple, il ne suffit pas de se connecter au réseau et d'y participer de temps en temps. Les nœuds ont une variété de fonctions selon leurs types, tels que la sauvegarde des enregistrements, historiques des transactions exécutées sur le réseau depuis sa création. Mais un nœud peut également vérifier les composants de la transaction. Par exemple, jusqu'à récemment, le réseau de nœuds bitcoin rejetait un bloc qui avait été vérifié par les mineurs car la récompense Bitcoin de ces derniers était supérieure à la norme actuelle de 12,5 BTC.
Par conséquent, c'est une responsabilité considérable de vérifier si les mineurs se conforment aux règles. Ceux-ci sont simples :

- Une transaction ne peut pas être réalisée deux fois.
- Les blocs et transactions doivent respecter les formats standards.
- Les récompenses des blocs doivent être précisément celles du consensus en cours.

Il existe différents types de nœuds.

2.4.5.1 Un nœud complet

Par exemple, lorsqu'un nœud réserve l'intégralité de la Blockchain. On dit qu'elle est pleine et il permet à de nouveaux nœuds de la télécharger (son registre) à leur tour. Mais ce n'est pas leur seule fonction, car comme évoqué plus haut, le nœud complet a pour rôle de vérifier si les blocs arrivants suivent le consensus du réseau. Dans le cas contraire d'empêcher leur ajout du fait de la Blockchain, en les rejetant. Il est à noter que même si tous les autres nœuds du réseau considèrent le bloc valide, un seul nœud détecte une anomalie pour effectuer le rejet.

2.4.5.2 Un nœud léger

Les nœuds légers peuvent également vérifier les transactions réseau sans télécharger ni mettre à jour l'intégralité de la Blockchain. Pour ce faire, ils utilisent une fonctionnalité appelée SPV (Simplified Payement Verification) pour vérifier spécifiquement certaines transactions. Ils se reposent sur des nœuds complets, qui leur permettent d'accéder aux données nécessaires aux vérifications et sont donc dépendant de ces derniers et de leur fiabilité. Ce système permet une participation au réseau plus accessible sans avoir à passer par un service centralisé.

2.4.5.3 Un Mineur

Les mineurs sont une partie importante du système de réseau Bitcoin. La vérification telle que décrite dans [64] d'une transaction est effectuée par le mineur. Ils sont également responsables de la vérification et de la validation des transactions, Ils participent au processus d'exploration, et créent de nouveaux blocs. Afin de vérifier la transaction, le mineur a fait deux choses. Ils reposent sur le fait que chaque transaction exécutée dans le système est copiée et mise à la disposition de n'importe quel pair du réseau. Ils vérifient également si le client expéditeur dispose de suffisamment d'argent pour initier la transaction. Une fois le solde de votre compte confirmé, le mineur générera une valeur de hachage. **Cette valeur de hachage doit avoir un format bien spécifique et doit commencer par plusieurs zéros**.

Les deux entrées qui sont nécessaires pour le calcul de la valeur de hash sont :

- Les données de l'enregistrement de la transaction ;
- La preuve de travail du mineur.

Lorsque la valeur appropriée pour la preuve de travail est trouvée par le mineur, il ou elle a récompensé des frais de transaction, qui peuvent être ajoutés à la transaction validée. La Blockchain, un format spécifique de base de données, contient chaque transaction validée qui a été transmise aux pairs. Ainsi cette valeur de hachage sera stockée comme mentionné dans les sections précédentes dans l'en-tête du bloc enfant (voir Figure *11*).

Après avoir évoqué l'importance des nœuds dans la technologie Blockchain. Nous allons voir dans la suite les différents types de Blockchain qui existent. Car l'utilisation des nœuds peut différer selon le contexte dans lequel on prévoit de les utiliser.

2.5 Les différents types de blockchain

2.5.1 La blockchain publique

Une Blockchain publique est un système de Blockchain qui présente les caractéristiques suivantes :

- Il dispose d'un réseau ouvert où les nœuds peuvent se joindre et partir à leur guise sans demander l'autorisation de qui que ce soit ;
- Tous les nœuds complets du réseau peuvent vérifier chaque nouvelle donnée ajoutée à la structure des données, y compris les blocs et les transactions ;
- Son protocole comprend un mécanisme incitatif qui vise à assurer le bon fonctionnement du système de Blockchain, y compris que les transactions valides sont traitées et incluses dans le grand livre et que les transactions invalides sont rejetées.

Ainsi nous pouvons donner l'exemple des deux plus grandes plateformes publiques de la technologie blockchain à savoir : Bitcoin et Ethereum.
Nous allons nous focaliser sur la plateforme Ethereum qui constitue notre domaine d'application dans le cadre de notre travail.

2.5.1.1 Blockchain Ethereum

2.5.1.1.1 Présentation du mode de fonctionnement du réseau Ethereum

Un réseau Blockchain se compose de plusieurs nœuds appartenant à des mineurs et de quelques nœuds qui ne sont pas des mineurs et de quelques nœuds qui ne font pas de minage mais qui aident à l'exécution de contrats intelligents et de transactions [65]. Ces derniers sont connus sous le nom

d'EVM. Chaque nœud est connecté à un autre nœud du réseau. Ces nœuds utilisent le protocole pair-à-pair pour communiquer entre eux.

Chaque mineur maintient une instance du grand livre. Un registre contient tous les blocs de la chaîne. Avec plusieurs mineurs, il est tout à fait possible que l'instance de registre de chaque mineur ait des blocs différents de ceux d'un autre. Les mineurs synchronisent leurs blocs en permanence pour s'assurer que l'instance de grand livre de chaque mineur est identique à celle de l'autre. Les détails concernant les grands livres, les blocs et les transactions sont abordés en détail dans les sections suivantes de ce chapitre. L'EVM héberge également des contrats intelligents. Les contrats intelligents permettent d'étendre Ethereum en y écrivant des fonctionnalités commerciales personnalisées. Une personne ayant un compte sur un réseau peut envoyer un message pour transférer de l'éther de son compte à un autre ou peut envoyer un message pour invoquer une fonction dans un contrat. Ethereum ne les distingue pas en ce qui concerne les transactions. La transaction doit être signée numériquement avec la clé privée d'un titulaire de compte [66]. Cela permet de s'assurer que l'identité de l'expéditeur peut être établie tout en vérifiant la transaction et en modifiant les soldes de plusieurs comptes. Jetons un coup d'œil aux composants d'Ethereum dans le diagramme suivant :

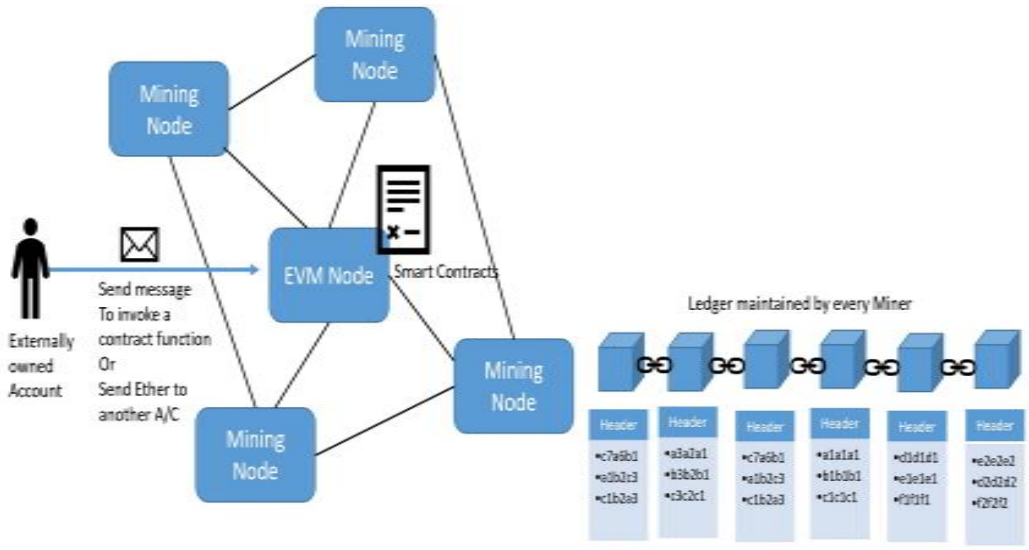


Figure 14 : Mode de fonctionnement d'un réseau Ethereum *[67]*

Les détails de la plateforme Ethereum et des smarts contracts sont mis dans l'annexe 1.

2.5.2 Une blockchain privée

Il s'agit de Blockchains tournant sur un réseau privé, dans lesquelles tous les participants sont connus et pour lesquels la gouvernance est assurée par une organisation. Personne ne peut y accéder et y participer sans y être autorisé.

2.5.3 Blockchain consortium

Il s'agit de Blockchains dans lesquelles le processus de consensus (validation des transactions /blocs) est contrôlé par un nombre connu et restreint de nœuds. Certains nœuds peuvent être rendus publics (accès autorisé en lecture seule) tandis que d'autres restent privés. Elles sont plus adaptées aux contextes régulés.

2.6 Les Algorithmes de consensus blockchain

L'algorithme de consensus est un mécanisme par lequel les nœuds du réseau Blockchain parviennent à un accord sur la validité et l'authenticité des transactions ou des blocs de données. Etant donné que les transactions Blockchain sont décentralisées. Le mécanisme de consensus est le processus central de vérification et de protection des blocs de transactions en faisant deux choses. En premier l'algorithme de consensus s'assure d'abord que le prochain bloc ajouté est la seule vraie version. Deuxièmement, l'algorithme peut empêcher tout adversaire de réussir à faire dérailler la chaîne. Tout système décentralisé, y compris la monnaie, doit résoudre le problème connu sous le nom de problème des généraux byzantins.

Les algorithmes de consensus sont le fondement de toutes les Blockchains. Ils sont la partie la plus importante des plateformes Blockchain. Sans eux (algorithmes de consensus), nous nous retrouverions avec une base de données stupides et immuable [68].

Alors des explications sur la question des généraux byzantins s'imposent.

Dans la question du général byzantin (BG), un groupe de généraux commandant l'armée byzantine encercle une ville. Si seulement quelques généraux attaquent la ville, l'attaque échouera. Les généraux doivent parvenir à un accord sur l'opportunité d'attaquer ou non par la communication. Cependant, il peut y avoir des traîtres parmi les généraux. Les traîtres peuvent envoyer différentes décisions à différents généraux. C'est un environnement sans confiance. Comment parvenir à un

consensus dans un tel environnement est un défi. Puisque le réseau blockchain est distribué, c'est aussi un défi pour la Blockchain. Dans la Blockchain, il n'y a pas de nœud central pour s'assurer que les registres sur les nœuds distribués sont les mêmes [69]. Le nœud n'a pas besoin de faire confiance à d'autres nœuds. Par conséquent, un certain protocole est nécessaire pour assurer la cohérence des registres des différents nœuds.

Nous avons constaté que ces algorithmes utilisent deux mécanismes à la base, à savoir la preuve ou le vote [70].

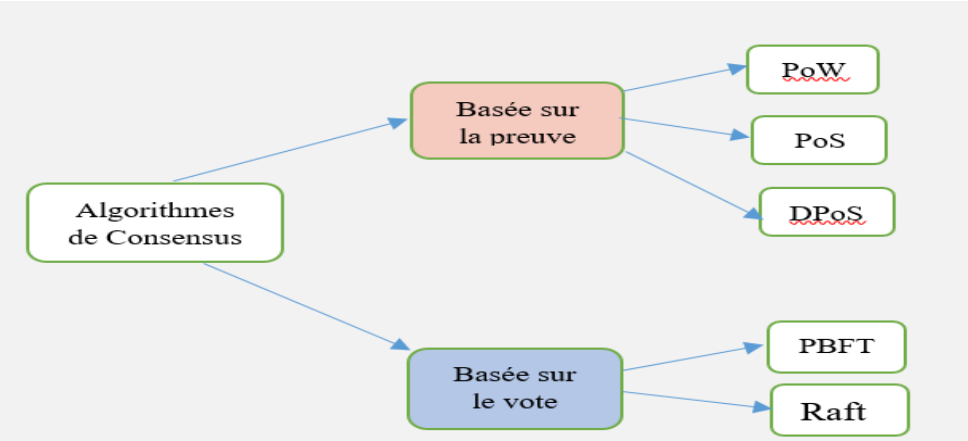


Figure 15: Classification des algorithmes de consensus

2.6.1 Les algorithmes de consensus basé sur la preuve

Comme nous avons eu à le détailler dans la section 2.4.2, les algorithmes de consensus basé sur la preuve a été introduit par Satoshi Nakamoto afin d'éviter la double dépense plus particulièrement le PoW. Au vu des inconvénients tel que la consommation d'énergie. Nous assistons à la naissance de la preuve d'enjeu PoS. Dans ce type de consensus les validateurs ou mineurs doivent posséder une quantité requise de pièces pendant un certain temps avant de participer au processus de validation. Cet algorithme est à l'abri de l'attaque des 51% et consomme moins d'énergie.

Parmi les solutions alternatives au PoW, nous pouvons citer le DPoS. Cet algorithme a proposé d'améliorer la sécurité du PoS, où l'élection des producteurs de blocs ou des témoins dépend des votes des parties prenantes. Cela offre les avantages du contrôle semi-central du réseau, tels que l'efficacité et la vitesse, tout en conservant les caractéristiques de la décentralisation.

2.6.2 Algorithme de consensus basé sur le vote

Les algorithmes de consensus basé sur le vote nécessitent l'identification des nœuds avant de commencer le travail de minage.

Ces algorithmes sont basés sur le vote fonctionnent comme les méthodes Tolerating Faults implémentées dans l'environnement distribué. Par ailleurs, ils peuvent résister aux pannes de crash ou à la subversion de certains nœuds. Ainsi ils peuvent être classé en deux catégories :

- Consensus byzantin basé sur la tolérance aux pannes : pourrait empêcher les deux situations à savoir les nœuds qui tombent en panne et les nœuds subvertis.
- Consensus basé sur la tolérance aux pannes de plantage : pourrait empêcher uniquement le cas de plantage des nœuds. Les sous-sections suivantes fourniront un résumé de deux des algorithmes de consensus basés sur le vote.

A titre d'exemple nous pouvons citer : PBFT (tolérance de panne byzantine pratique) et Raft.

2.7 Les domaines d'application de la technologie blockchain

La Blockchain a d'abord été utilisée dans les crypto-monnaies où son succès a été constaté à partir du Bitcoin. Aujourd'hui, il existe de nombreux domaines d'applications de cette technologie.

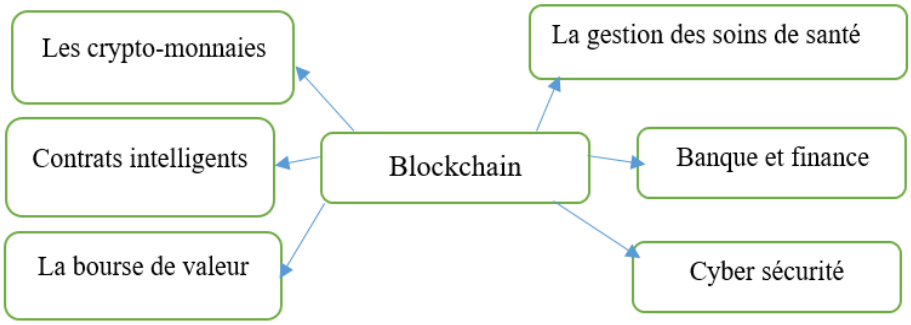


Figure 16 : Les domaines d'application de la technologie blockchain

2.7.1 Les crypto-monnaies

La technologie Blockchain est à la base du bitcoin et de nombreuses autres crypto-monnaies comme l'Éther avec des capitalisations boursières assez élevée. Il existe actuellement environ 1200 crypto-monnaies, dont le Bitcoin-cash, le Litecoin, le Dash, le Ripple, le Monero, Zcash, et

plusieurs autres [71]. De nombreuses entreprises et vendeurs acceptent les paiements en crypto-monnaies.

2.7.2 Les contrats intelligents

Le smart contract est un contrat régi et exécuté par un programme informatique sans avoir besoin d'un tiers tel qu'un avocat. Le programme exécute automatiquement les accords contractuels de manière équitable lorsque ses conditions sont satisfaites. Avec le soutien de la Blockchain, les contrats intelligents sont actuellement sûrs et pratiques [72] [73] [74].

2.7.3 La bourse de valeur

Les méthodes traditionnelles d'achat et de vente d'actifs et d'actions nécessitent beaucoup de coûts indésirables, de fiducies (contrats par lequel un bien est cédé comme garantie à un créancier, lequel devra le restituer au débiteur lorsque celui-ci aura rempli ses obligations) et d'implications d'intermédiaires. Avec la technologie Blockchain, ces frais généraux pourraient être surmontés. L'éminente transformation du marketing boursier par la technologie Blockchain a été envisagée par Microsoft [75]. Bitshares, Augur, NASDAQ et Coinsetters (ce sont des bourses de valeur) ont utilisé la Blockchain pour la commercialisation des actions et les échanges [76] [77].

2.7.4 La gestion des soins de santé

Le système actuel de gestion des soins de santé présente plusieurs problèmes tels que l'incohérence des données, la duplication des dossiers et l'incapacité des patients de connaître et de gérer leurs propres dossiers. La Blockchain, lorsqu'elle est utilisée correctement pourrait résoudre ces problèmes de santé. La Blockchain est actuellement utilisée pour partager et sécuriser les données pour la gestion des soins de santé. L'Estonie est le premier gouvernement à mettre ses dossiers de santé sur la Blockchain [78] [79] [80].

2.7.5 Banque et finance

Blockchain est capable de perturber l'industrie bancaire et financière. De nombreuses banques ont essayé la Blockchain pour améliorer leurs systèmes. La première transaction bancaire avec Blockchain a été réalisée en 2016 entre la Commonwealth Bank of Australia et Wells Fargo [81]. Ils ont étudié les défauts du système bancaire et ont mis en évidence des solutions utilisant la

Blockchain. Plusieurs autres services financiers comme les paiements en ligne et les actifs numériques sont réalisés avec Blockchain [82].

2.7.6 Cyber sécurité

La Blockchain est utilisée pour améliorer la cyber sécurité. L'historique du réseau, la configuration, les fichiers journaux et d'autres fichiers réseau sont stockés dans le réseau Blockchain pour fournir des enregistrements sécurisés et immuables contre les attaquants. Ce concept est utilisé par des entreprises comme Guardtime pour fournir des services de sécurité réseau contre plusieurs attaques réseau [83] [84] [85].

2.8 Quelle est la base de la confiance en la technologie blockchain ?

De par ces caractéristiques citées dans la section 2.2 , la technologie Blockchain dispose des atouts suivants pour instaurer la confiance et telle que mentionnée dans [86] :

- **Architecture décentralisée** : elle est basée sur un grand nombre de nœuds, dépendant de diverses organisations. Cela signifie qu'avec une architecture centralisée où les décisions peuvent être prises unilatéralement, qu'il s'agisse d'un consensus ou d'un contrôle réussi de plus de 50 % des nœuds (ou de la puissance de calcul) est nécessaire pour avoir un impact sur le système. Le fait que l'architecture repose sur de nombreux nœuds qui assurent la vérification et le stockage de la Blockchain garantit également une meilleure disponibilité du service.
- **Mécanisme incitatif attractif :** le nombre de nœuds doit être suffisamment important et issu d'organisations différentes pour garantir l'indépendance et la disponibilité de la Blockchain. Les mineurs bénéficient d'un gain après chaque opération de minage.
- **Traçabilité et audibilité de toute la chaîne de transaction :** exposer toutes les transactions effectuées depuis l'origine de la Blockchain (bloc 0 ou Genesis bloc) dans l'espace public, permettant à chacun de **vérifier l'intégrité** de la chaîne et suivre toutes les actions liées au compte. Il n'y a donc plus de possibilités de tricherie, tout se voit, tout se sait, dans les garanties du pseudonyme.
- **Transparence algorithmique** : Le code utilisé pour miner, interagir avec la Blockchain ou mettre en place des smarts contracts est lisible par tout le monde. L'avantage est d'avoir des

experts de la communauté. L'utilisateur scrute le code et déclenche une alerte en cas de doute. La confiance repose donc sur le lanceur d'alerte.

- **Authenticité de transaction protégée par un pseudonyme** : Les transactions doivent être approuvées par le titulaire du compte à l'aide d'un matériel **cryptographique suffisamment sécurisé** pour éviter que des commandes ne soient passées à son insu. Afin de s'adapter aux avancées technologiques, il est important de fournir des mécanismes avec un niveau de sécurité adaptatif.
- **Blockchain rigide avec des garanties de sécurité importantes** : Il est nécessaire de rigidifier les blocs contenus dans la Blockchain et l'ordre de ces blocs pour éviter toute modification ultérieure de la Blockchain [87]. Pour cela, il faut s'appuyer sur le caractère distribué de l'architecture, et s'appuyer sur un puissant mécanisme de consensus. À cela, nous pouvons ajouter un mécanisme pour encourager les bons comportements, un mécanisme pour décourager les mauvais comportements et fournir du matériel cryptographique support technique solide. PoW repose sur un consensus et des preuves cryptographiques coûteuses en calcul, tandis que PoS repose sur un consensus et des mécanismes d'incitation et de suppression qui n'ont pas été prouvés sur des systèmes réels. En réalité, le Proof of Stake n'a pas encore été déployé sur la plateforme Ethereum.

2.9 Risques et limites

- **Gouvernance neutre :** Avant d'investir du temps et de l'argent dans la Blockchain, il faut se poser les questions suivantes : La gouvernance est-elle garantie ?

  Acteurs impliqués, c'est-à-dire petits groupes ou les personnes impliquées dans l'élaboration du code et du protocole sont-elles véritablement indépendantes dans la prise de décision ?

  Peuvent-elles résister à la pression politique, industrielle ? Si ce n'est pas le cas, le principe de base de la décentralisation n'est plus respecté.

  Non respecté pour les raisons suivantes. Lorsque le code Blockchain est mis à jour avec les nouvelles règles de fonctionnement, Blockchain, les administrateurs de mineurs peuvent choisir d'accepter ou de refuser les mises à jour. Il peut s'agir d'une transformation de règle

mineure et rétro compatible, lorsqu'il s'agit de "soft fork" - ou de changements majeurs sans compatibilité ascendante, là nous parlerons de "hard fork".

Pour devenir pratique, un "soft fork" nécessite le soutien d'une majorité de mineurs, tandis qu'un "hard fork" la bifurcation nécessite le soutien d'un consensus plus large.

- **Neutralité de l'infrastructure informatique sous-jacente** : l'allocation des ressources informatiques utiles au calcul et au stockage dans la Blockchain doit être équilibrée entre les organisations. La tendance de bitcoin à créer des pools de minage a conduit à plusieurs reprises à ce que les trois plus grandes fermes mutualisent plus de 50% de la puissance du réseau. Ainsi s'ils mutualisent plus de 51% de la puissance de calcul, ils pouront controler l'historique de la blockchain.
- **Erreur de programmation** : Pour les Blockchains programmables (ou non), le risque élevé est associé aux erreurs humaines de programmation, comme ce fut le cas en 2016 Utilisé dans une attaque de détournement en arrière-plan sur Ethereum "The DAO" (The DAO for Decentralized Autonomous Organizations). Ce bogue permet aux escrocs de boucler indéfiniment les fonctions qui conduisent à la sortie. En 2017, une autre attaque impliquait un bogue dans le logiciel du portefeuille Parity Wallet et a entraîné le vol de 30 millions de dollars en éther.
- **La double dépense** : elle consiste à enregistrer deux transactions liées au même objet et s'excluent généralement mutuellement. Il s'agit d'un acte malveillant délibéré de la part d'un participant, généralement lors du processus de minage, l'une des transactions sera rejetée. Cependant, lors d'une scission de chaîne, il peut arriver que chaque transaction soit vérifiée indépendamment par chaque chaîne. A ce moment, le bénéficiaire sait s'il a un bonus jusqu'à ce que la chaîne la plus courte soit abandonnée. Pour Bitcoin, un délai raisonnable à considérer est d'environ 1 heure, ce qui équivaut à 6 blocs.
- **Retenir les transactions** : les mineurs peuvent être intéressés à ne pas partager les transactions avec des frais de transaction élevés. En conservant cette transaction pour lui jusqu'à ce qu'un bloc soit miné avec succès, il s'assure que la transaction soit incluse dans l'un de ses blocs et qu'il sera récompensé. Par conséquent, cela peut prendre beaucoup de temps. De telles attaques de rétention deviendront de plus en plus crédibles à l'avenir, car les modèles de paiement s'appuieront de plus en plus sur les frais de transaction à mesure que les récompenses globales diminuent.

- **Blanchiment d'argent** : Chaque fois que des problèmes de blanchiment d'argent surviennent, de nouvelles méthodes d'échange sont disponibles. Contrairement à ce que l'on pourrait penser, la transparence des transactions dans la blockchain n'empêche pas le blanchiment d'argent, elle ne fait que le compliquer. En effet, des technologies existent pour couvrir les trajectoires et la traçabilité. La première est très simple et consiste à détenir plusieurs comptes (dont certains ne peuvent être utilisés qu'une seule fois) et à commercer entre plusieurs d'entre eux. Une autre façon de perturber la traçabilité, connue dans bitcoin sous le nom de Coinjoin, consiste à combiner plusieurs transactions en une seule. Plus le nombre d'opérations de fusion est élevé (entrant et sortant) est important, meilleure est la protection car plus il est difficile de mettre en relation débiteurs et créanciers. Mais notez que cette méthode Zerocash garantit la non traçabilité des transactions. Ce qui rend impossible la détermination ou la détection du blanchiment d'argent sur la seule base des éléments fournie dans la Blockchain.

2.10 Les attaques contre la technologie Blockchain

La technologie Blockchain est explorée dans de nombreuses applications innovantes. Grâce à l'architecture pair à pair transparente et totalement distribuée de la Blockchain, ces applications bénéficient d'un modèle d'annexe seulement dans lequel les transactions acceptées dans la Blockchain ne peuvent être modifiées [88]. La transparence de la Blockchain permet de stocker des enregistrements publiquement vérifiables et indéniables. En outre, le système de pair à pair de la Blockchain permet la tenue d'un grand livre vérifiable sans autorité centralisée, ce qui résout le problème du point unique de défaillance et du point unique de confiance.

Tirant parti de ces propriétés, plusieurs applications Blockchain ont été développées, notamment des crypto-monnaies comme le Bitcoin et le Litecoin, des plateformes de contrats intelligents comme Ethereum et des organisations autonomes décentralisées (DAO) comme Dash et Bitshares. Malgré les caractéristiques fonctionnelles que la Blockchain apporte à l'espace de conception de ces applications, des rapports récents ont souligné les risques de sécurité associés à cette technologie. Ces risques sont surtout dus à des pertes d'argents (ce qui constitue un des principaux mobiles pour commettre leurs forfaits). Par ailleurs, ses attaques peuvent être motivées par le fait d'amener les utilisateurs à se tourner vers d'autres crypto-monnaies. En raison de leur nature publiquement vérifiable, les crypto-monnaies basées sur la Blockchain sont vulnérables à plusieurs activités frauduleuses. Car il faut souligner que l'utilisation de la chaîne de bloc ne se limite pas

seulement aux crypto-monnaies et aux contrats intelligents. Ces systèmes à grande échelle contiennent les données personnelles des utilisateurs (Par exemple : les applications des soins de santé). C'est pourquoi la sécurité de ces systèmes constitue un élément indispensable pour leur acceptation par le grand public.

A cela s'ajoute le fait que certains services de sécurité ont été mis en péril dans la technologie Blockchain par des attaques les plus puissantes : déni de service distribué (DDoS) [89], l'attaque par collision [90], attaques par centralisation [91] [92] [93] , la sybille, éclipse, attaques par injection de code, attaques par rejeu et attaques par ransomware. Nous allons nous focaliser sur les attaques qu'a subi la technologie Blockchain suivant différents domaines [94].

Nous présentons le résumé de ces attaques dans le tableau ci-dessous

Attaques à différent niveau	Attaque basée sur le hachage	Attaque à 51%
		Attaque par collision
	Attaque par centralization	Exploitation minière égoïste
		Attaque par bourrage de scrutin
	Attaque par le traffic	Attaque DDoS
		Attaque par usurpation de message
	Attaque au niveau du réseau	Attaque Sybil
		Attaque Eclipse
	Injection ou attaque de l'initié	Attaque par injection de code
		Injection SQL
		Attaque par injection de fautes
	Attaque d'intégrité	Attaque par falsification

		Attaque par logiciel malveillant : • Ransomware • Cryptojacking
	Attaque par fuite de clé privée	Attaque de l'homme du milieu
		Attaque par clé
		Attaque par relecture

Tableau 4 : Résumé des attaques contre la Blockchain

2.11 Conclusion

En définitive, nous avons passé en revue l'ensemble des notions essentielles qui permet de mieux appréhender la technologie Blockchain. Ce qui permet de dire que la Blockchain est une technologie prometteuse qui présente d'immenses avantages. Parmi lesquels nous pouvons citer la sécurité des données, la réduction des coûts, l'anonymat, la rapidité, la transparence, la traçabilité et, surtout, l'éviction des intermédiaires et des autorités centrales. Elle apporte une révolution numérique en bouleversant de nombreux secteurs. Actuellement, il existe de nombreuses applications de la Blockchain en dehors des crypto-monnaies, ainsi que plusieurs adoptions dans de nombreux pays et entreprises. Ce qui permet d'envisager dans un futur proche d'autre adoption, car la technologie aura gagné en termes de maturité.

On pense que la Blockchain sera finalement adoptée par le grand public dans le monde entier.

Cependant il faudra régler un certain nombre de questions en ce qui concerne l'évolutivité de la technologie, la confidentialité et l'intégrité des données.

Sans oublier l'aspect le plus important et indispensable de la technologie Blockchain est la sécurité.

C'est pourquoi le chapitre qui suit présente une discussion générale sur la technologie Blockchain.

Un accent particulier sera mis sur les fonctions de hachages et la sécurité des blockchain qui intègre les smarts contracts.

Chapitre III : Discussion et Perspectives de la Blockchain avec Ethereum comme domaine d'application

Discussion et Perspectives de la Blockchain et Ethereum comme domaine d'application

La blockchain fonctionne dans un environnement décentralisé qui est rendu possible par la présence de plusieurs technologies de base, telles que les signatures numériques, les hachages cryptographiques et les algorithmes de consensus distribués. Toutes les transactions ont lieu de manière décentralisée, ce qui élimine la nécessité de recourir à un ou des intermédiaires pour valider et vérifier les transactions.

Elle couvre plusieurs domaines d'application et devra faire face à un certain nombre de défis pour atteindre son apogée.

C'est dans cadre que nous présentons notre discussion qui montre à quel point il est fondamental de résoudre ces problèmes de sécurité et de mise en place de fonction de hachage plus performant qui assurerait l'intégrité des données.

Ainsi après avoir présenté les challenges d'évolutivité et de confidentialité notre discussion se concentrera sur deux niveaux à savoir :

- **Au niveau des fonctions de hachage :** Elles constituent une plaque tournante de la technologie blockchain. Nous allons présenter la problématique liée aux fonctions de hachage. Puis rappelait les constructions des fonctions de hachage qui sont les plus souvent utilisées. Puis terminer par émettre l'idée de mettre en place de nouvelle fonction de hachage dont la construction sera basée sur celles connues.

- **Au niveau des smarts contracts** : à ce stade nous allons nous focaliser sur la sécurité des smart contract et évoquer les outils qui permettent d'atténuer les failles de sécurité liées au smart contract. Car ces failles de sécurité sont liées aux smart contrats qui constituent aujourd'hui une des plus grandes menaces pour la technologie Blockchain. Enfin nous allons évoquer la possibilité de mettre en place un **débogueur de smart contract à base de l'intelligence artificielle**.

3 Challenge de la technologie Blockchain

La technologie Blockchain a un énorme potentiel et promet de révolutionner le monde. Mais pour être pleinement performantes, les Blockchains doivent surmonter plusieurs défis : évolutivité, confidentialité et de sécurité.

3.1 Evolutivité

L'évolutivité est une préoccupation majeure pour les plates-formes basées sur la blockchain [95]. Dans le cas de Bitcoin, la taille et la fréquence limitées des blocs, ainsi que le nombre de transactions que le réseau peut gérer, peuvent entraîner des problèmes d'évolutivité. Comparé à Paypal et Visa qui traitent 200 et 2000 transactions par seconde. L'augmentation de la taille du bloc est la considération la plus importante ; c'est ce qui à entrainer l'adoption d'une solution en 2017 après que la blockchain Bitcoin ait été submergée. Une taille de bloc a été augmentée de 1 Mo à 2 Mo, ce qui a conduit à un fork car une très petite communauté avait décidé que 2 Mo n'étaient pas suffisants et que 8 Mo étaient nécessaires pour éviter d'éventuelles surcharges futures. Il existe une autre solution pour réduire le temps de vérification des blocs d'envoi. Comme Ethereum, le temps de vérification des blocs est de 15 secondes, tandis que Bitcoin est de 10 minutes.

Des solutions ont été proposées pour alléger la surcharge du réseau. Ces solutions externalisent certaines tâches du réseau. Puis compare les sorties de serveur et suppose qu'il y a au moins un nœud honnête.

L'idée de la mise en place du Bitcoin nouvelle génération a été émise aussi (préciser la source). Cette idée consiste à diviser le bloc en deux : d'une part nous aurons les blocs clés pour l'élection du chef et le micro bloc destiné au stockage des transactions. Les mineurs entrent en concurrence pour devenir le chef qui générera les micros blocs jusqu'à ce que le chef soit désigné.

3.2 Confidentialité

Elle fait partie de l'une des propriétés rechercher dans la technologie blockchain surtout avec les Blockchains publiques ou l'ensemble des transactions sont connues de tous. Il faut cependant souligner que ceux qui émettent ces transactions ne divulguent pas leurs identités personnelles. Les utilisateurs s'envoient leurs transactions par le biais d'une adresse publique. Ce dernier permet d'assurer l'anonymat des personnes impliquées dans le processus. Cependant il faut noter qu'il est

impossible d'assurer la confidentialité des transactions dans la mesure où elles sont connues de tous. De ce fait, les membres du réseau sont au courant des changements de solde de chaque clé publique ou adresse. De plus, Biryukov et al. [96] ont proposé une méthode pour lier un pseudonyme d'utilisateur à une adresse IP qui pourrait révéler l'identité du participant. Par conséquent, ce manque de confidentialité entrave le déploiement de Blockchains publiques dans divers domaines où la confidentialité est critique, tels que la finance, la santé, l'entreprise, etc. Aussi, il convient de noter qu'il existe plusieurs techniques pour assurer la confidentialité des personnes, telles que : l'offuscation non identifiable, l'utilisation de cryptage homomorphe, les preuves à connaissance zéro. On rappelle que l'obscurcissement indiscernable pourrait servir de solution pour la problématique de la confidentialité dans les plateformes Blockchains. Cependant elle n'a pas encore été déployé. Elle deviendrait un mécanisme d'obscurcissement incassable qui deviendrait un contrat intelligent dans une boite noire.

- Le cryptage homomorphe

Elle consiste à effectuer des opérations sur des données cryptées sans en connaître le contenu. En résumé, les données sont envoyées à un serveur cloud. Puis après traitement elles seront renvoyées sans pour autant que leur intégrité ne soit remise en question. Ainsi une fois implémenté dans la Blockchain, elle pourrait servir à assurer la confidentialité des transactions.

- La preuve de connaissance zéro ou ZKP

C'est un protocole qui permet de prouver qu'un utilisateur est à l'origine d'une transaction sans qu'il n'est besoin de prouver son identité ou divulguer des informations personnelles.

3.3 Sécurité

3.3.1 Les problématiques liées aux fonctions de hachage

L'une des principales motivations de cette discussion au niveau des fonctions de hachage découle de l'importance et le rôle prépondérant qu'ils jouent pour le bon fonctionnement de la technologie Blockchain. Nous la retrouvons un peu partout dans la technologie mais plus particulièrement au niveau de la couche applicative et une fois que les données sont hachées, elle fait partie des artefacts qui sont générées. Plus particulièrement après compilation des smarts contracts au niveau de la machine virtuelle de Ethereum.

A cela s'ajoutent, le fait que les fonctions de hachage SHA-2 (256, 384 et 512) sont basées sur la construction de Merkle Damgard et celle de Keccak sur la construction éponge.
Cela constitue un avantage, dans la mesure où si un jour on parvient à casser l'une cela ne va pas impliquer la cassure de l'autre.
Cependant avec l'évolution et des prouesses technologiques qui s'enchaînent, nous ne sommes pas à l'abri d'une mauvaise surprise. Compte tenu de l'environnement décentralisé, il est nécessaire d'empêcher des attaquant de revenir sur des transactions.

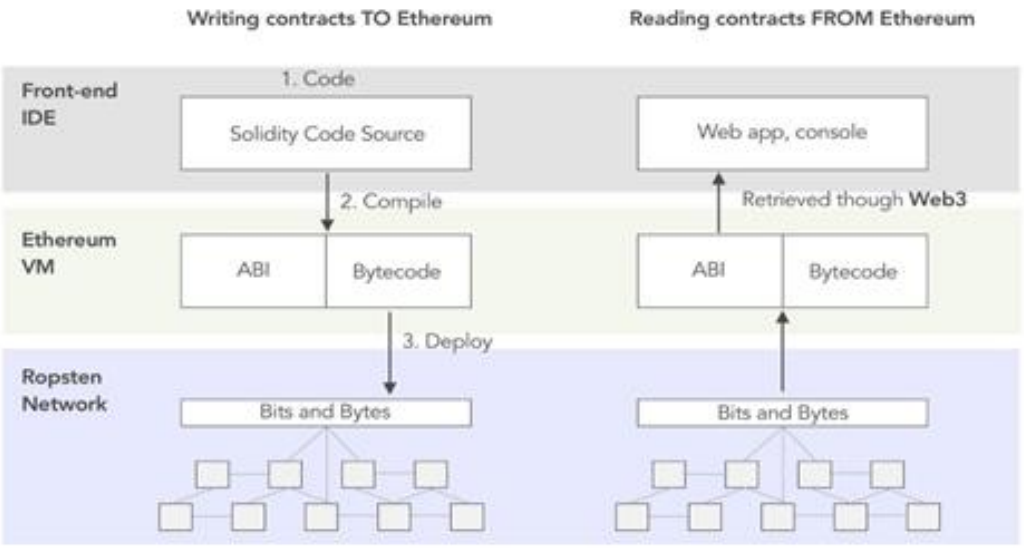


Figure 17 : Processus de compilation et de déploiement d'un contrat intelligent*[97]*

D'où la nécessité d'avoir une fonction de hachage fiable et qui est infaillible face aux attaques génériques.
Ainsi nous allons présenter les différentes constructions des fonctions de hachage et qui reste infaillible.

3.3.1.1 Rappels des constructions des fonctions de hachage

3.3.1.1.1 Construction de Merkle-Damgard

Pour les structures Merkle-Damguard, le message d'entrée est divisé en morceaux une taille fixe, disons 512 bits, puis nous choisissons un vecteur d'initialisation (VI) comme clé de chiffrement symétrique qui est de la même taille que le hachage de 256 bits que nous voulons générer dans ce cas. Le premier bloc de 512 bits est ensuite chiffré à l'aide de la fonction de compression C fournie

par l'algorithme de chiffrement par bloc (DES ou AES). Le résultat de cette fonction est de 256 bits et une opération XOR (OU exclusif) est appliquée entre le résultat et la clé (VI). Le résultat du XOR est utilisé pour chiffrer le bloc suivant avec la fonction de compression C, et le même processus est répété jusqu'à ce que tous les blocs aient été traités. Cette figure décrit la méthode de Merkle-Damgard.

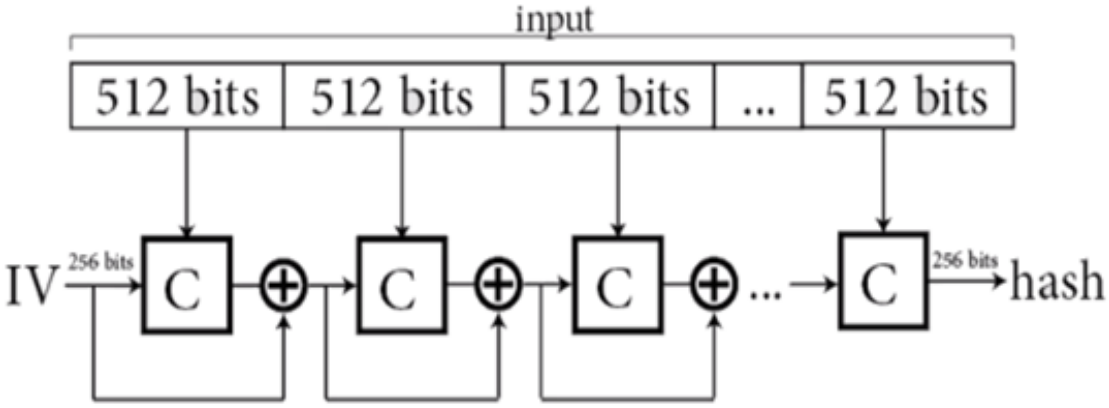


Figure 18 : Construction Merkle-Damgard *[98]*

3.3.1.1.2 Construction HAIFA de la fonction Blake-2

Il faut savoir que HAÏFA est un Framework qui est basé sur la construction de Merkle-Damgard. Cependant il corrige un certain nombre de failles qui ont récemment été découvertes dans la construction de Merkle-Damgard ou des fonctions dont la construction de base est Merkle-Damgard, parmi lesquelles nous pouvons citer : MD5 et SHA-1. Ces failles permettent de réaliser les attaques de pré-image et de second pré-image. Même lorsque les fonctions de compressions sous-jacentes sont sécurisées.

Ainsi HAIFA permet de corriger de nombreuses failles tout en supportant plusieurs propriétés supplémentaires **tel que l'ajout d'un compteur bit ou valeur de sel au niveau l'entrée de chaque fonction de compression.** Ce qui permet de contrecarrer l'attaque générique de la pré-image et propose des tailles variables de valeur haché. HAIFA permet aussi un calcul en ligne de la fonction en un seul passage avec une quantité fixe de mémoire, indépendamment de la taille du message [99].

3.3.1.1.3 Construction de Keccak

La construction de Keccak, également connue sous le nom de construction de l'éponge, se compose de deux étapes. Commencez par absorber les informations avant d'extraire le hachage. La première

étape consiste à définir deux bits, r et c, puis à diviser le message d'entrée en blocs de r bits. Initialiser un vecteur de longueur r+c bits, puis appliquer XOR entre les r premiers bits du vecteur et le premier bloc du message d'entrée. Le résultat du XOR avec les c bits du vecteur initial est donné en entrée de la fonction de permutation aléatoire f. Cette fonction fournit deux sorties, une de r bits et une de c bits ; XOR est appliqué à la sortie de r bits et au deuxième bloc du message d'entrée. Envoyez à nouveau le résultat à la fonction f et ainsi de suite jusqu'à ce que tous les blocs soient traités.

La deuxième étape commence par ajouter le résultat des r bits de la fonction f sur le dernier bloc à la chaîne binaire (hash).

À chaque itération de f, nous ajoutons r bits du résultat au hachage jusqu'à obtenir un hachage de la taille souhaitée, disons 256 bits.

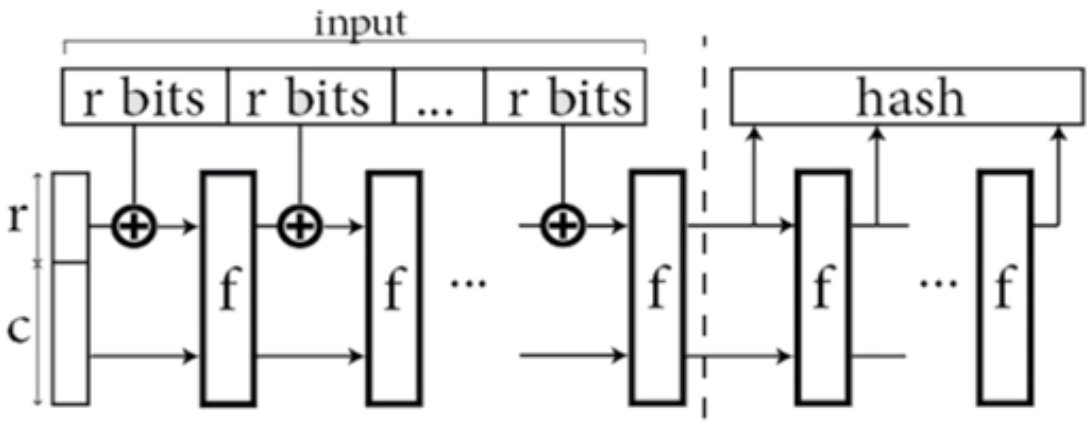


Figure 19 : construction keccak

Au moment où nous rédigeons ce mémoire, les fonctions de hachage basée sur les constructions cités ci-dessus ne sont pas remis en question ou ne présentent de bonnes caractéristiques face aux attaques génériques.

Par conséquent des efforts continuent d'être fournie dans les recherches afin de trouver des fonctions plus puissantes. Qui présenteraient des caractéristiques supérieures à celles connus actuellement.

3.3.2 Les problématiques liées au smarts contracts

Malgré cette popularité croissante, le développement de contrats intelligents reste encore un peu un mystère pour de nombreux développeurs, en grande partie à cause de sa conception et de ses applications particulières.

Les smarts contracts **peuvent gérer de grosses sommes d'argent, des actifs numériques, des stocks ou des données**. Il est très important de tenir compte de l'aspect sécurité des contrats intelligents, car même un minuscule bug peut entraîner des problèmes importants. Par exemple l'attaque du DAO en juin 2016 (implique une perte de 60 millions de dollars USD).

Ainsi l'attaquant a appelé récursivement la fonction DAO divisée pour transférer de l'Ether (crypto-monnaie Ethereum) vers son propre compte et les appels se sont arrêtés avant de mettre à jour le nouveau solde du contrat appelant (le compte de l'attaquant). L'écriture de contrats intelligents sûrs et exempts de bogues est une tâche difficile, car les études précédentes montrent qu'un pourcentage important de contrats intelligents déjà déployés sur la blockchain Ethereum sont vulnérables.

Le tableau ci-dessous présente les attaques qui impliquent les smarts contracts dans différentes plateformes blockchain.

3.3.2.1 Attaque incriminant les contrats intelligents

Dans le tableau ci-dessous, nous donnons un résumé des attaques qui incriminent les contrats intelligents [100].

Nom de l'attaque	Date de l'attaque	Nom du contrat	Perte
Attaque de la DAO	06/17/2016	DAO	60 million de dollars US
Parité multi-signature	07/19/2017	Parité multi-signature	31 million de dollars US
Parité multi-signature	11/06/2017	Parité multi-signature	280 million de dollars US

BECToken	04/22/2018	ERC-20	Réduit à une valeur nulle
Débordement de proxy	04/26/2018	SMT	16 millions de TMS perdus
Débordement de lot	05/24/2018	EDU	Réduit à une valeur nulle
N/A	05/24/2018	BAI	Réduit à une valeur nulle
N/A	06/16/2018	ICX	Tous les ICX gelés
Lendf.Me	04/19/2020	DeFi	25 MILLION DE DOLLARS US
Gouverner l'esprit	2017	Gouverner l'esprit	Réduit à une valeur nulle
Attaques de Fomo3D	Juillet 2018	Fomo3D	3 MILLION DE DOLLARS US

Tableau 5 : Les attaques incriminants les contrats intelligents dans les plateformes Blockchains

Nous constatons que les contrats intelligents font face à plusieurs qui risque d'influencer les investisseurs à s'intéresser à ce domaine et ébranlent la confiance du grand public quant à l'adoption de cette dernière.

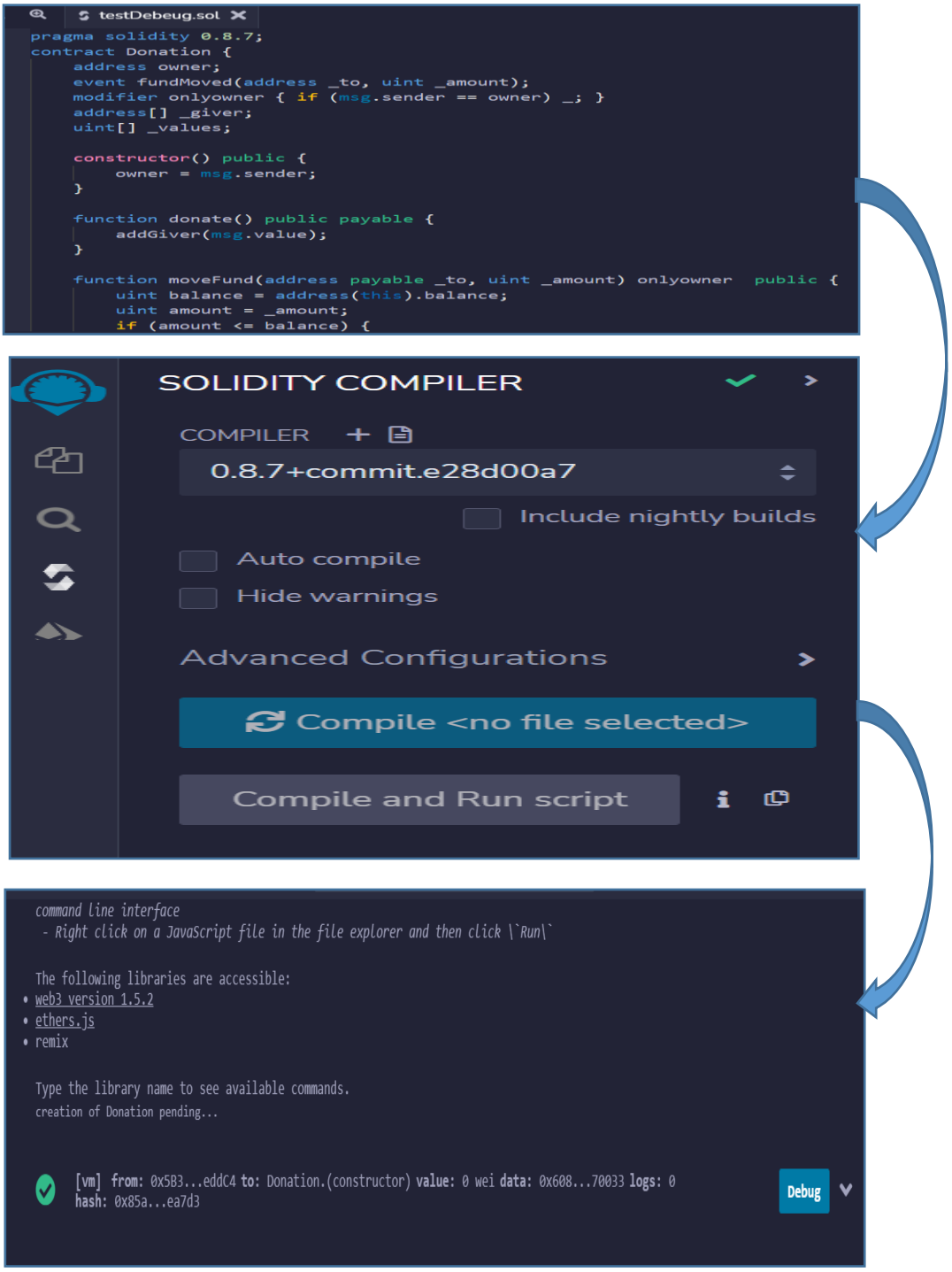


Figure 20 : Processus de compilation d'un smart contract *[101]*

Vous pouvez constater sur la dernière image le débogueur qui valide le code de smart contract saisi.

Par la même occasion vous pouvez lire le haché du code du smart contract concerné.

3.3.2.2 Etude de comparaison des plateformes blockchain intégrant les contrats intelligents

Comme stipulé dans le tableau ci-dessous, nous avons plusieurs plateformes Blockchains qui intègrent les contrats intelligents. Cependant il faut constater que ces plateformes n'ont pas le même mode de fonctionnement dans la mesure où elles intègrent différentes machines virtuelles en leurs sein et utilisent des fonctions de hachages différentes (au sein de ces machines virtuelles afin de hacher les données).

Plateformes Blockchains	Utilisent les Crypto-monnaies	Utilisent une Machine Virtuelle	Les fonctions de Hachages utilisées	Langages de programmation
Bitcoin	Bitcoin	Native	SHA-256	Ivy,RSK, BitML
Ethereum	Ether	EVM	Keccak -256	Solidity, Flint, SCILLA
NEO	NEO	NeoVM	SHA-256	C#,VB.Net, F#,Java,Kotlin, Python
Cardano	ADA	IELE	SHA-512	Plutus (langages fonctionnels
EOS	EOS	WebAssembly (WASM)	SHA-256	C++
Solana	SOL	LLVM	SHA-256	Rust, C [7]
Tezos	XTZ	Tezos VM	BLAKE2 et SHA-512	Michelson

Tableau 6 : Les plateformes Blockchains qui intègrent les contrats intelligents et les fonctions de hachages utilisées au niveau des machines virtuelles

Ainsi nous pouvons classer les problèmes de sécurité des smarts contract peuvent être classé comme suit :

- Une dépendance de l'ordre des transactions

Dans la blockchain Ethereum, l'ordre d'exécution des transactions dépend des mineurs et les clients n'ont aucun contrôle sur cet ordre. Ce problème survient lorsque plus d'une transaction est invoquée par le même contrat, et l'ordre de ces transactions peut affecter le nouvel état de la blockchain. Les auteurs suggèrent la condition de garde comme solution. L'idée est que la confirmation de la transaction devient dépendante de la satisfaction de la condition de garde, sinon la transaction est abandonnée.

- Une dépendance à l'horodatage

Ce problème est lié aux contrats intelligents, qui comprennent des conditions qui se déclenchent par l'horodatage des blocs. Les horodateurs de bloc sont définis par les mineurs en fonction de l'heure de leur système local et peuvent donc être manipulés par un adversaire. Les auteurs suggèrent d'utiliser l'index des blocs au lieu de l'horodatage des blocs, car il est incrémentiel et les protège de toute manipulation.

- Des exceptions mal gérées

Ce problème vise les contrats qui appellent un autre contrat. Si une exception se produit dans un smart contract appelé, celui-ci se termine et renvoie **false**, mais il peut ne pas en informer le smart contract appelant. La suggestion du document pour ce problème est d'ajouter des instructions EVM explicites **throw** et **catch**.

- Vulnérabilité de réentrainement

Ce problème est lié à la vulnérabilité des DAO (expliquée au début de la section 3). Lorsqu'un contrat appelle un autre contrat, l'exécution du contrat en cours attend que le contrat appelé se termine. Cela donne l'opportunité à l'adversaire d'exploiter l'état intermédiaire du contrat appelant et d'appeler ses méthodes plusieurs fois.

C'est dans ce contexte que des outils ont été conçu pour atténuer la sécurité des smarts contracts.

- ***Outils d'analyse de la sécurité des smarts contracts***

Méthode	Exemples
vérification formelle	F*, KEVM

Outils d'analyse du code des contrats intelligents	OYENTE, SECURIFY
Langage de programmation sécurisé	Bambo, SCILLA, Flint
Connexion sécurisée entre la blockchain et le hors-chaîne	Town Crier, Smart Cast

Tableau 7 : d'outils d'analyse de smart contract *[102]*

Comme le montre le tableau ci-dessus, il existe d'abord des outils qui sont basés sur la vérification formelle des smarts contracts. Ces outils sont basés sur le calcul formel et permettent de vérifier à travers un modèle mathématique afin de vérifier les bytecode de smart contract (KEVM). Alors que F* est un outil qui permet de prendre en entrée le code du smart contract et le code compilé du smart contract afin de passer à la vérification formelle.

Ensuite nous avons les outils d'analyse de codes de smart contract comme son nom l'indique, permettent de vérifier le code du smart contract et d'en détecter les failles. Ainsi l'outil OYENTE a permis de détecter 45% de smart contract non sécurisé sur 19366 smart contract sur la plateforme Ethereum.

L'analyse de Securify consiste en deux étapes. Tout d'abord, elle analyse symboliquement le graphe de dépendance du contrat pour extraire des informations sémantiques précises du code. Ensuite, il vérifie les configurations de conformité et de violation qui capturent les conditions suffisantes pour prouver qu'une propriété est valide ou non. Pour permettre l'extensibilité, tous les modèles sont spécifiés dans un langage spécifique au domaine. Securify est diffusé publiquement, il a analysé plus de 18 000 contrats soumis par ses utilisateurs et est régulièrement utilisé pour effectuer des audits de sécurité par des experts.

Puis en ce qui concerne les langages de programmations sécurisés, ils imposent un mode de fonctionnement et l'état de la machine virtuelle. Ce qui ne permet pas d'avoir une très grande flexibilité. Car nous vivons dans un environnement décentralisé et sans confiance.

Enfin en ce concerne les outils de connexion sécurisé entre la blockchain et le hors-chaîne, ils jouent le rôle de pont entre les smarts contracts et les sites web existants qui sont déjà couramment fiables pour les applications non-blockchain. Il combine un frontend blockchain avec un backend matériel de confiance pour gratter les sites web compatibles HTTPS et servir des données authentifiées à la source aux contrats intelligents. TC prend également en charge la confidentialité. Il permet des demandes de données privées avec des paramètres chiffrés.

Cependant des interrogations subsistent toujours malgré que des efforts et outils aient été mis en place. Au vu de tout ce qui a été dit ci-dessus, nous tirons des petites conclusions suivantes :

- Il n'y a pas de moyen efficace de garantir la sécurité du code des contrats intelligents ;
- Les outils de développement existants sont encore très rudimentaires ;
- Il n'y a pas d'outils de gestion de la sécurité ;
- Les langages de programmation et les machines virtuelles présentent encore un certain nombre de limitations ;
- Les problèmes de performance sont difficiles à gérer dans un contexte de ressources limitées. Les problèmes de performance sont difficiles à gérer dans un environnement où les ressources sont limitées (y compris les documents avancés/mis à jour et la communauté).

Car il y une exigence très élevée en matière de sécurité de code des smarts contracts. Ainsi il est difficile de garantir la sécurité de ces derniers dans un tel environnement (pour l'instant).

Une meilleure pratique serait de favoriser **les tests unitaires, les tests d'intégration et enfin les tests de performance** des smarts contracts.

3.3.3 Proposition de mise en place de débogueur intelligent

Au vu de tout ce qui a été dit précédemment, nous proposons la mise en place d'un smart contract à base d'apprentissage automatique.

L'idée sera de mettre en place un débogueur qui se distingue des débogueurs actuels qui sont utilisés au niveau des plateformes en ligne.

Le schéma ci-dessous décrit de manière général le concept :

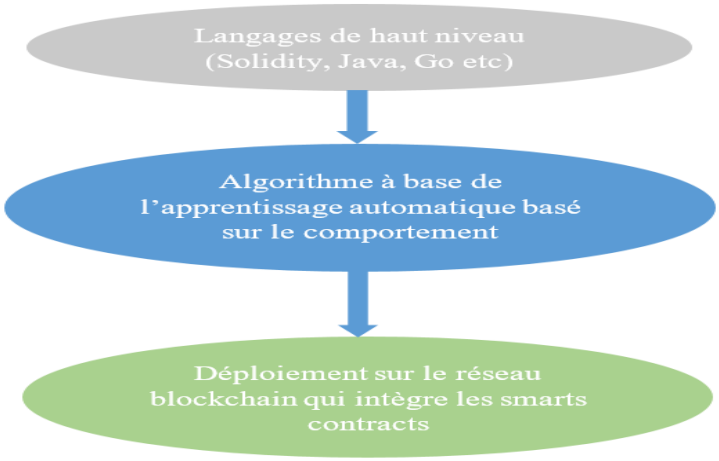


Figure 21 : Exemple de débogueur à base d'apprentissage automatique

Comme illustré sur le schéma, le débogueur sera un outil qui sera construit à base de l'intelligence artificielle.

Par ailleurs, il faut signaler qu'il existe des débogueurs qui sont misent en ligne. Cependant ces derniers ne corrigent que des failles liées à la **syntaxe** et la **sémantique** des smarts contrats au niveau du compilateur.

Ainsi ces débogueurs n'ont pas la possibilité de prendre en compte les failles existantes dans les smarts contracts.

C'est dans ce contexte que nous préconisons la mise en place d'un débogueur qui suit les principes de ***l'apprentissage automatique ou machine learning***.

Nous rappelons que l'objectif de l'intelligence artificielle est de créer des entités intelligentes [103].

L'apprentissage automatique ou machine learning est une forme d'intelligence artificielle qui a pour but d'avoir un système qui s'améliore par l'expérience. On en distingue deux types à savoir :

- La mémorisation par cœur implique de mémoriser explicitement tous les exemples possibles pouvoir les restituer ;
- L'apprentissage par généralisation vise à extraire des règles implicites d'un grand nombre d'exemples afin de les appliquer à de nouvelles situations rencontrées. Pour les machines conditionnelles, la mémorisation par cœur est relativement facile. En revanche, l'apprentissage par généralisation est difficile car il nécessite d'extraire des règles qui ne

sont pas explicitement mentionnées dans les exemples. Ce qui constitue un défi pour ce domaine.

C'est sur la base de cette deuxième méthode que nous allons établir notre débogueur. Ainsi nous allons faire apprendre à notre algorithme de débogage les attaques qui ont déjà été menées avec succès. Par ailleurs, à chaque qu'une autre attaque sera menée, alors elle sera prise en compte. Nous allons définir un certain nombre de seuil que notre débogueur. Ce qui implique le fait que si un smart contract s'éloigne de ce seuil, elle sera considérée comme malveillante, par contre si elle se rapproche alors elle sera considérée comme normale ou acceptée.

Elle servira de modèle de base pour mettre en place dans l'avenir des débogueurs plus puissant afin de permettre à la technologie d'atteindre son apogée.

Elle pourra aider à éviter les pertes financières et accroître la confiance du public en la technologie. Dans la mesure où les attaques qui ont été mentionnées ci-dessus, ont contribué à briser cette confiance.

L'une des causes de ces attaques est due au fait que les outils qui sont utilisés pour le développement de ces smarts contracts sont encore récents. Par exemple Solidity.

Il faut souligner que des solutions pareilles existent, c'est-à-dire des outils qui sont basés sur l'apprentissage automatique tel que SoliAudi. Mais ce dernier se concentre sur l'ABI et le bytecode déployé sur le réseau.

Ainsi même pour un développeur aguerri dans le domaine du développement de logiciel standard trouve du mal à coder un smart contract qui ne comporte pas de vulnérabilité de sécurité. Ce qui montre qu'il n'est pas évident de comprendre le mode de fonctionnement des smarts contracts et le niveau de difficulté et des exigences que cela peut avoir.

3.3.4 Comparaison des algorithmes de consensus

La sécurité des blockchains dépend de la robustesse et de la force de l'algorithme de consensus qui est utilisé pour vérifier les transactions et les blocs.

Ainsi le temps de vérification et de validation d'une transaction a un impact sur le système car c'est ce processus qui déterminera le temps nécessaire à la génération d'un nouveau bloc.

La mesure de la transaction par seconde est utilisée pour calculer la performance des systèmes qui traitent les transactions de routine. La transaction par seconde est utilisée pour déterminer la vitesse de la plate-forme ou du réseau dans l'exécution des transactions. Plus le nombre de transactions par seconde est élevé, plus les transactions seront exécutées, validées et confirmées rapidement sur la même plateforme [104].

Par exemple, si une crypto-monnaie exécute 12 transactions par minute, le temps est de 0,2. Cela nous indique que si cette crypto-monnaie est capable d'effectuer 12 transactions en 60 secondes, le temps de réponse pour chaque transaction sera de 20 secondes. Ainsi, la transaction par seconde est un critère important dans le réseau blockchain et dépend de son algorithme de consensus. Les blockchain d'aujourd'hui sont conçues pour effectuer des TPS élevés [105].

Algorithme de consensus	Fonction de hachage	Plateformes de crypto-occurrence	Nombre de Transaction/seconde	Temps de création d'un bloc (en minutes)
PoW	SHA-256	Bitcoin	7	10
	Ethash (Keccak-256)	Ethereum	15	0,25
PoS	SHA-256	Nxt	100	1
	Blake-2	Nano	7000	Instantanément
DPoS	Ouroboros	EOS	4000	0,5
	DPoS	Cardano	257	0,33
PBFT	Keccak	Zilliqa	0	45 s to 4 minutes
	N/A	Ripple	1500	0,06
Dbft	RIPEMD160	NEO	1000	0,25
PoA (hybride PoS et PoW)	Blake-256	Decred	14	5
	SHA-256	Peercoin	0	10

Tableau 8 : Comparaison des algorithmes de consensus en fonction de leurs crypto-monnaies représentatives

Il faut savoir que ce temps de vérification des transactions influe sur le temps de génération ou de création de bloc. En d'autres termes, c'est le temps qui s'écoule entre le moment où une valeur est présentée au réseau et celui où elle fait l'objet d'un consensus dans le réseau.

Il existe également un autre facteur qui affecte le processus de validation dans les réseaux blockchain, appelé **bootstrap time**, qui signifie le temps qu'il faut à un nouveau nœud pour télécharger et traiter l'historique nécessaire pour valider l'état actuel du système. Par ailleurs, ce facteur dépend fortement de la puissance de calcul du nœud connecté au réseau ou à la plateforme.

A cela s'ajoute le temps de confirmation et de validation d'une transaction. Nous rappelons qu'une transaction entre l'expéditeur et le destinataire est dite valide dans une blockchain si elle est demandée par l'expéditeur. Lorsqu'un utilisateur effectue une transaction, il doit utiliser sa clé privée comme signature numérique. Lorsqu'une transaction devient valide, elle est incluse dans un bloc et ce bloc est ajouté au réseau blockchain. Lorsqu'une transaction est incluse dans un bloc miné, elle reçoit une confirmation. Si le nombre de blocs minés est plus élevé, le nombre de confirmations augmentera pour la transaction. En effet, pour réduire le risque d'une attaque par double dépense, un certain nombre de confirmations de blocs sont nécessaires.

Une fois minée, le mineur reçoit une récompense en contrepartie. Différents facteurs influent sur la rentabilité du minage, notamment la complexité du processus, les récompenses du minage, la consommation d'énergie, les frais de transaction et la dépendance du processus de minage à l'égard d'un matériel spécifique. Les algorithmes de consensus permettent de parvenir à un sur les valeurs des données entre plusieurs nœuds dans un système distribué.

Des solutions ont été proposées mais jusqu'à présent nous, la communauté n'a pas réussi à mettre en place un algorithme de consensus qui soit exempt de tout reproche.
C'est dans ce contexte que nous proposons une optimisation des temps de transaction. Les données étant envoyées sur le net afin qu'elles puissent atteindre tous les nœuds du réseau. L'utilisation d'algorithmes de plus court chemin serait peut-être idéale afin de réduire le temps vérification et de validation des transactions et par la même occasion celui de la création de bloc.

3.4 Conclusion

Comme nous l'avons eu à l'évoquer tout au long de ce chapitre, après avoir montré l'importance de l'évolutivité et de la confidentialité dans la technologie Blockchain. Nous avons parlé de la sécurité surtout des plateformes qui intègrent les smarts contracts dans la technologie Blockchain.

A cela s'ajoutent les propositions de solution Blockchain en ce qui concerne les nouvelles constructions des fonctions de hachage et surtout celle qui concerne l'architecture de vérification des smart contracts.

Conclusion Générale

En définitive, dans ce travail de mémoire qui nous a permis de mettre en évidence la sécurité qui sous-tend la technologie dans le chapitre I. Puis nous avons fait l'état de l'art dans le chapitre II. Pour finir par montrer les bases de la confiance, ainsi que les risques et limites de cette dernière. Nous avons terminé par le chapitre III, avec une discussion et des propositions de solution.

La technologie Blockchain continue de révolutionner plusieurs industries et secteurs de recherches. Ainsi à partir de 2015, l'avènement des contrats intelligents ouvre de nouvelles perspectives. Car la popularité croissante et l'application approfondie de la technologie blockchain, les smart contract émergents sont devenus un point névralgique de la recherche dans les universités et l'industrie. La nature décentralisée, exécutoire et vérifiable des contrats intelligents permet aux termes du contrat d'être appliqués entre des parties non fiables sans l'implication d'une autorité de confiance ou d'un serveur central. Par conséquent, les contrats intelligents devraient révolutionner de nombreuses industries traditionnelles telles que la finance, la gestion, l'IoT, etc.

Les contrats intelligents basés sur la blockchain passent du battage médiatique au déploiement dans le monde réel. Comme il est utilisé pour prendre en charge des transactions financières de grande valeur, il est essentiel d'assurer sa sécurité afin d'assurer une adoption par le grand public.

La sécurité des contrats intelligents doit être assurée pour éviter les pertes inutiles et les attaques malveillantes. Ainsi nous avons présenté d'abord quelques-unes de ces attaques au niveau du chapitre II. C'est dans cette suite logique que nous avons évoqué les attaques qui ont été menées contre la plateforme Ethereum. Nous rappelons que ce dernier est la première plateforme à avoir intégré les contrats intelligents.

Ensuite, nous avons terminé par présenter les attaques impliquant les smarts contracts dans les différentes plateformes Blockchains.

Puis dans ce travail, nous avons aussi parlé du niveau de sécurité de la technologie Blockchain en tant que telle et évoqué la complexité des fonctions de hachage avant de réaliser une étude de comparaisons des fonctions de hachages qui jouent un rôle indispensable dans le bon fonctionnement de la technologie blockchain.

Nous avons terminé par effectuer de mettre en évidence l'importance de l'ABI dans la machine virtuelle de Ethereum. Mettre en valeur son importance dans la jonction entre les deux couches à savoir applicative et réseau.

Plusieurs mécanismes d'analyse sont mis en œuvre pour tester et garantir l'exactitude et l'invulnérabilité des modèles dans les contrats intelligents. Mais les développeurs et les utilisateurs de contrats intelligents doivent être conscients de la précision et des performances de ces méthodes analytiques. Il faut préciser que ce n'est pas la technologie en tant que telle qui est remise en question. Cependant la plupart des attaques exploitent les failles de programmation. C'est dans ce contexte que des outils ou logiciels à base de calcul formel ont été mis en place. En guise de perspective, nous préconisons la mise en place d'un environnement de test de contrat intelligent. Définir un mode de développement standard afin d'éviter les attaques impliquant les contrats intelligents.

Références

[1] M. AbuNaser et A. A. Alkhatib, « Advanced survey of blockchain for the internet of things smart home », in *2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT)*, 2019, p. 58–62.

[2] P. Qian, Z. Liu, Q. He, B. Huang, D. Tian, et X. Wang, « Smart contract vulnerability detection technique: A survey », *ArXiv Prepr. ArXiv220905872*, 2022.

[3] S.-Y. Lin, L. Zhang, J. Li, L. Ji, et Y. Sun, « A survey of application research based on blockchain smart contract », *Wirel. Netw.*, vol. 28, nº 2, p. 635-690, févr. 2022.

[4] M. B. Yassein, S. Aljawarneh, E. Qawasmeh, W. Mardini, et Y. Khamayseh, « Comprehensive study of symmetric key and asymmetric key encryption algorithms », in *2017 international conference on engineering and technology (ICET)*, 2017, p. 1–7.

[5] « Full Text ». .

[6] « « PKI ». http://igm.univ-mlv.fr/~dr/XPOSE2007/vma_PKI/c... - Google Scholar ». [En ligne]. Disponible sur: https://scholar.google.com/scholar?hl=fr&as_sdt=0%2C5&q=%C2%AB+PKI+%C2%BB.+http%3A%2F%2Figm.univ-mlv.fr%2F%7Edr%2FXPOSE2007%2Fvma_PKI%2Fconcepts_de_base.html+%28consult%C3%A9+le+22+juin+2022%29.&btnG=. [Consulté le: 10-nov-2022].

[7] J. Treger, « Etude de la sécurité de schémas de chiffrement par bloc et de schémas multivariés », PhD Thesis, Versailles-St Quentin en Yvelines, 2010.

[8] M. A. RAKOTOMALALA, T. E. RAKOTONDRAINA, et S. R. RAKOTONDRAMANANA, « Transmission sécurisée d'image utilisant un chiffrement par bloc combine avec la transformée d'Arnold », *Afr. Sci.*, vol. 14, nº 3, p. 323–335, 2018.

[9] A. K. Mandal, C. Parakash, et A. Tiwari, « Performance evaluation of cryptographic algorithms: DES and AES », in *2012 IEEE Students' Conference on Electrical, Electronics and Computer Science*, 2012, p. 1–5.

[10] A. K. Mandal, C. Parakash, et A. Tiwari, « Performance evaluation of cryptographic algorithms: DES and AES », in *2012 IEEE Students' Conference on Electrical, Electronics and Computer Science*, 2012, p. 1–5.

[11] H. Diakonoff, R. Felizardo, H. Thomas, et C. Clément, « Mise au point sur les possibilités d'usage de la blockchain en médecine légale », *Rev. Médecine Légale*, vol. 13, nº 1, p. 23–29, 2022.

[12] V. Autefage et D. Magoni, « WireGuard ou comment mettre en place un VPN en un temps record », in *JRES 2022: 14èmes Journées Réseaux de l'Enseignement et de la Recherche*, 2022, vol. 14, p. Article–30.

[13] S. Pattanayak et S. A. Ludwig, « Encryption based on neural cryptography », in *International Conference on Hybrid Intelligent Systems*, 2017, p. 321–330.

[14] Y. B. Kim, T.-Y. Youn, et S. C. Seo, « Chaining optimization methodology: a new sha-3 implementation on low-end microcontrollers », *Sustainability*, vol. 13, nº 8, p. 4324, 2021.

[15] A. Maetouq, S. Daud, N. Ahmad, N. Maarop, N. N. A. Sjarif, et H. Abas, « Comparison of hash function algorithms against attacks: A review », *Int. J. Adv. Comput. Sci. Appl. Br*, vol. 8, 2018.

[16] A. Maetouq, S. Mohd, N. Azurati, N. Maarop, N. Nur, et H. Abas, « Comparison of Hash Function Algorithms Against Attacks: A Review », *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, nº 8, 2018.

[17] M. A. Kale et S. Dhamdhere, « Survey Paper on Different Type of Hashing Algorithm », vol. 3, nº 2, p. 3, 2018.
[18] T. Li, Y. Sun, M. Liao, et D. Wang, « Preimage attacks on the round-reduced Keccak with cross-linear structures », *IACR Trans. Symmetric Cryptol.*, p. 39–57, 2017.
[19] P.-L. Cayrel, G. Hoffmann, et M. Schneider, « GPU implementation of the Keccak hash function family », in *International Conference on Information Security and Assurance*, 2011, p. 33–42.
[20] C. Paar et J. Pelzl, « Sha-3 and the hash function keccak », *Underst. Cryptogr.- Textb. Stud. Pract.*, 2010.
[21] J.-P. Aumasson, S. Neves, Z. Wilcox-O'Hearn, et C. Winnerlein, « BLAKE2: simpler, smaller, fast as MD5 », in *International Conference on Applied Cryptography and Network Security*, 2013, p. 119–135.
[22] J.-P. Aumasson, L. Henzen, W. Meier, et R. C.-W. Phan, « Sha-3 proposal blake », *Submiss. NIST*, vol. 92, 2008.
[23] L. Henzen, J.-P. Aumasson, W. Meier, et R. C.-W. Phan, « VLSI characterization of the cryptographic hash function BLAKE », *IEEE Trans. Very Large Scale Integr. Vlsi Syst.*, vol. 19, nº 10, p. 1746–1754, 2010.
[24] P. Pritzker et P. D. Gallagher, « Sha-3 standard: permutation-based hash and extendable-output functions », *Inf. Tech Lab. Natl. Inst. Stand. Technol.*, p. 1–35, 2014.
[25] K. M. AbdulMajeed, « Hash functions: analysis study », Master's Thesis, Kuala Lumpur: International Islamic University Malaysia, 2013, 2013.
[26] « Full Text ». .
[27] « Full Text ». .
[28] « Full Text ». .
[29] C. Boura, « Analyse de fonctions de hachage cryptographiques », PhD Thesis, Université Pierre et Marie Curie-Paris VI, 2012.
[30] A. I. Sanka, M. Irfan, I. Huang, et R. C. Cheung, « A survey of breakthrough in blockchain technology: Adoptions, applications, challenges and future research », *Comput. Commun.*, vol. 169, p. 179–201, 2021.
[31] A. D. Alrehily, A. F. Alotaibi, S. B. Almutairy, M. S. Alqhtani, et J. Kar, « Conventional and improved digital signature scheme: A comparative study », *J. Inf. Secur.*, vol. 6, nº 01, p. 59, 2015.
[32] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, et H. Wang, « Blockchain challenges and opportunities: A survey », *Int. J. Web Grid Serv.*, vol. 14, nº 4, p. 352–375, 2018.
[33] A. D. Alrehily, A. F. Alotaibi, S. B. Almutairy, M. S. Alqhtani, et J. Kar, « Conventional and improved digital signature scheme: A comparative study », *J. Inf. Secur.*, vol. 6, nº 01, p. 59, 2015.
[34] D. Toradmalle, R. Singh, H. Shastri, N. Naik, et V. Panchidi, « Prominence of ECDSA over RSA digital signature algorithm », in *2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on*, 2018, p. 253–257.
[35] « Full Text ». .
[36] « Full Text ». .
[37] O. D. Alowolodu, B. K. Alese, A. O. Adetunmbi, O. S. Adewale, et O. S. Ogundele, « Elliptic curve cryptography for securing cloud computing applications », *Int. J. Comput. Appl.*, vol. 66, nº 23, 2013.
[38] « Full Text ». .

[39] H. Hasanova, U. Baek, M. Shin, K. Cho, et M.-S. Kim, « A survey on blockchain cybersecurity vulnerabilities and possible countermeasures », *Int. J. Netw. Manag.*, vol. 29, nº 2, p. e2060, 2019.
[40] B. Bhushan, P. Sinha, K. M. Sagayam, et J. Andrew, « Untangling blockchain technology: A survey on state of the art, security threats, privacy services, applications and future research directions », *Comput. Electr. Eng.*, vol. 90, p. 106897, 2021.
[41] « Full Text ». .
[42] H. Han, S. Fei, Z. Yan, et X. Zhou, « A survey on blockchain-based integrity auditing for cloud data », *Digit. Commun. Netw.*, 2022.
[43] U. Padmavathi et N. Rajagopalan, « Concept of blockchain technology and its emergence », in *Research Anthology on Convergence of Blockchain, Internet of Things, and Security*, IGI global, 2023, p. 21–36.
[44] « Full Text ». .
[45] « Full Text ». .
[46] W. Dai, « B-money », *Consulted*, vol. 1, nº 2012, p. 412, 1998.
[47] H. Finney, « Rpow-reusable proofs of work », *Agosto 15*, 2004.
[48] « Full Text ». .
[49] « Full Text ». .
[50] J. Paris, « Apports des Smart Contracts aux Blockchains et comment créer une nouvelle crypto-monnaie », PhD Thesis, Haute école de gestion de Genève, 2017.
[51] M. Kaur, M. Z. Khan, S. Gupta, A. Noorwali, C. Chakraborty, et S. K. Pani, « MBCP: Performance analysis of large scale mainstream blockchain consensus protocols », *IEEE Access*, vol. 9, p. 80931–80944, 2021.
[52] « Full Text ». .
[53] D. Ongaro, *Consensus: Bridging theory and practice*. Stanford University, 2014.
[54] « Full Text ». .
[55] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, et S. Capkun, « On the security and performance of proof of work blockchains », in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, p. 3–16.
[56] « Full Text ». .
[57] « LA BLOCKCHAIN ENFIN EXPLIQUÉE ». [En ligne]. Disponible sur: https://gouvernance.news/2018/06/20/la-blockchain-enfin-expliquee/. [Consulté le: 11-nov-2022].
[58] « Qu'est-ce que la double-dépense ? — Bitpanda Academy ». [En ligne]. Disponible sur: https://www.bitpanda.com/academy/fr/lecons/quest-ce-que-la-double-depense/. [Consulté le: 11-nov-2022].
[59] A. Bakhoum, « La Blockchain pour la sécurisation des E-livrets scolaires. », 2019.
[60] « Comprendre la blockchain Ethereum - Article 1 : Bitcoin, première implémentation de la blockchain (1/2) | Ethereum France », 03-juin-2016. .
[61] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, et H. Wang, « Blockchain challenges and opportunities: A survey », *Int. J. Web Grid Serv.*, vol. 14, nº 4, p. 352–375, 2018.
[62] Y. K. Tomov, « Bitcoin: Evolution of blockchain technology », in *2019 IEEE XXVIII International Scientific Conference Electronics (ET)*, 2019, p. 1–4.
[63] Y. Lu, « Blockchain: A survey on functions, applications and open issues », *J. Ind. Integr. Manag.*, vol. 3, nº 04, p. 1850015, 2018.
[64] G. Wood, « Ethereum: A secure decentralised generalised transaction ledger », *Ethereum Proj. Yellow Pap.*, vol. 151, nº 2014, p. 1–32, 2014.

[65] R. Modi, *Solidity Programming Essentials: A beginner's guide to build smart contracts for Ethereum and blockchain*. Packt Publishing Ltd, 2018.
[66] P. Wackerow, S. Richards, et R. Cordell, « Ethereum development documentation », *Dostupno Na Httpsethereum Orgendevelopersdocs1 Lipanj 2021*, 2021.
[67] S. A. Bragadeesh, S. M. Narendran, et A. Umamakeswari, « Securing the Internet of Things Using Blockchain », in *Essential Enterprise Blockchain Concepts and Applications*, Auerbach Publications, 2021, p. 103–122.
[68] A. A. Monrat, O. Schelén, et K. Andersson, « A survey of blockchain from the perspectives of applications, challenges, and opportunities », *IEEE Access*, vol. 7, p. 117134–117151, 2019.
[69] S. M. H. Bamakan, A. Motavali, et A. B. Bondarti, « A survey of blockchain consensus algorithms performance evaluation criteria », *Expert Syst. Appl.*, vol. 154, p. 113385, 2020.
[70] A. I. Sanka et R. C. Cheung, « A systematic review of blockchain scalability: Issues, solutions, analysis and future research », *J. Netw. Comput. Appl.*, vol. 195, p. 103232, 2021.
[71] M. Bartoletti et L. Pompianu, « An empirical analysis of smart contracts: platforms, applications, and design patterns », in *International conference on financial cryptography and data security*, 2017, p. 494–509.
[72] A. Pinna, S. Ibba, G. Baralla, R. Tonelli, et M. Marchesi, « A massive analysis of ethereum smart contracts empirical study and code metrics », *IEEE Access*, vol. 7, p. 78194–78213, 2019.
[73] A. Kosba, A. Miller, E. Shi, Z. Wen, et C. Papamanthou, « Hawk: The blockchain model of cryptography and privacy-preserving smart contracts », in *2016 IEEE symposium on security and privacy (SP)*, 2016, p. 839–858.
[74] N. Biedrzycki, « Will blockchain transform the stock market », *Data Driven Invest.*, 2019.
[75] A. Almatarneh, « Blockchain technology and corporate governance: The issue of smart contracts—current perspectives and evolving concerns », *Éthique Économique Ethics Econ.*, vol. 17, n° 1, 2020.
[76] O.-B. Kwame *et al.*, « V-chain: A blockchain-based car lease platform », in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2018, p. 1317–1325.
[77] A. Corso, « Performance analysis of proof-of-elapsed-time (poet) consensus in the sawtooth blockchain framework », PhD Thesis, University of Oregon, 2019.
[78] Y. Sheng *et al.*, « Towards good correlation between fragment molecular orbital interaction energies and experimental IC50 for ligand binding: A case study of p38 MAP kinase », *Comput. Struct. Biotechnol. J.*, vol. 16, p. 421–434, 2018.
[79] T. McGhin, K.-K. R. Choo, C. Z. Liu, et D. He, « Blockchain in healthcare applications: Research challenges and opportunities », *J. Netw. Comput. Appl.*, vol. 135, p. 62–75, 2019.
[80] P. Gomber, « Hinz-O. Nofer M. Schiereck D.,'Blockchain' », *Springer*, vol. 59, n° 3, p. 183–187, 2017.
[81] Z. Kakushadze et J. A. Serur, « 151 Trading Strategies », *Z Kakushadze JA Serur*, vol. 151, 2018.
[82] T. Salman, M. Zolanvari, A. Erbad, R. Jain, et M. Samaka, « Security services using blockchains: A state of the art survey », *IEEE Commun. Surv. Tutor.*, vol. 21, n° 1, p. 858–880, 2018.
[83] J. Bouckaert, T. Van Dijk, et F. Verboven, « Access regulation, competition, and broadband penetration: An international study », *Telecommun. Policy*, vol. 34, n° 11, p. 661–671, 2010.

[84] A. I. Sanka, M. Irfan, I. Huang, et R. C. Cheung, « A survey of breakthrough in blockchain technology: Adoptions, applications, challenges and future research », *Comput. Commun.*, vol. 169, p. 179–201, 2021.
[85] M. Laurent, « La blockchain est-elle une technologie de confiance ». Institut Mines-Télécom, 2018.
[86] M. PIGNEL et D. STOKKINK, « LA TECHNOLOGIE BLOCKCHAIN Une opportunité pour l'économie sociale? » JUIN, 2018.
[87] D. Dasgupta, J. M. Shrein, et K. D. Gupta, « A survey of blockchain from security perspective », *J. Bank. Financ. Technol.*, vol. 3, nº 1, p. 1–17, 2019.
[88] R. Chaganti, B. Bhushan, et V. Ravi, « The role of Blockchain in DDoS attacks mitigation: techniques, open challenges and future directions », *ArXiv Prepr. ArXiv220203617*, 2022.
[89] S. Yaji, K. Bangera, et B. Neelima, « Privacy preserving in blockchain based on partial homomorphic encryption system for AI applications », in *2018 IEEE 25th International Conference on High Performance Computing Workshops (HiPCW)*, 2018, p. 81–85.
[90] S. Solat et M. Potop-Butucaru, « Brief announcement: Zeroblock: Timestamp-free prevention of block-withholding attack in bitcoin », in *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, 2017, p. 356–360.
[91] A. Sapirshtein, Y. Sompolinsky, et A. Zohar, « Optimal selfish mining strategies in bitcoin », in *International Conference on Financial Cryptography and Data Security*, 2016, p. 515–532.
[92] J. Willemson, « Bits or paper: Which should get to carry your vote? », *J. Inf. Secur. Appl.*, vol. 38, p. 124–131, 2018.
[93] Y. Wen, F. Lu, Y. Liu, et X. Huang, « Attacks and countermeasures on blockchains: A survey from layering perspective », *Comput. Netw.*, vol. 191, p. 107978, 2021.
[94] J. van den Hooff, M. F. Kaashoek, et N. Zeldovich, « Versum: Verifiable computations over large public logs », in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014, p. 1304–1316.
[95] A. Biryukov, D. Khovratovich, et I. Pustogarov, « Deanonymisation of clients in Bitcoin P2P network », in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014, p. 15–29.
[96] « What is EVM(Ethereum Virtual Machine)? | by eiki | Medium ». [En ligne]. Disponible sur: https://medium.com/@eiki1212/what-is-evm-ethereum-virtual-machine-f38310130114. [Consulté le: 12-nov-2022].
[97] H. Tiwari, « Merkle-Damgård construction method and alternatives: a review », *J. Inf. Organ. Sci.*, vol. 41, nº 2, p. 283–304, 2017.
[98] I. Al Shaikhli, M. Alahmad, et K. Munthir, « Hash function of finalist SHA-3: Analysis study », *Int. J. Adv. Comput. Sci. Inf. Technol. IJACSIT Vol*, vol. 2, p. 1–12, 2014.
[99] M. Ndiaye et P. K. Konate, « Cryptocurrency Crime: Behaviors of Malicious Smart Contracts in Blockchain », in *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, 2021, p. 1–8.
[100] « Remix - Ethereum IDE ». [En ligne]. Disponible sur: https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js. [Consulté le: 12-nov-2022].
[101] J.-W. Liao, T.-T. Tsai, C.-K. He, et C.-W. Tien, « Soliaudit: Smart contract vulnerability assessment based on machine learning and fuzz testing », in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, 2019, p. 458–465.

[102] L. D. Matteis, « Introduction à l'apprentissage », p. 18.
[103] X. Fu, H. Wang, et P. Shi, « A survey of Blockchain consensus algorithms: mechanism, design and applications », *Sci. China Inf. Sci.*, vol. 64, nº 2, p. 1–15, 2021.
[104] N. Six, N. Herbaut, et C. Salinesi, « Quelle Blockchain choisir? Un outil d'aide à la décision pour guider le choix de technologie Blockchain », in *INFORSID 2020*, 2020, p. 135–150.
[105] D. Lin, J. Wu, Q. Yuan, et Z. Zheng, « Modeling and Understanding Ethereum Transaction Records via a Complex Network Approach », *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 67, nº 11, p. 2737-2741, nov. 2020.
[106] « Ethereum development documentation », *ethereum.org*. [En ligne]. Disponible sur: https://ethereum.org. [Consulté le: 23-juin-2022].
[107] R. Modi, *Solidity Programming Essentials: A beginner's guide to build smart contracts for Ethereum and blockchain*. Packt Publishing Ltd, 2018.
[108] V. Y. Kemmoe, W. Stone, J. Kim, D. Kim, et J. Son, « Recent advances in smart contracts: A technical overview and state of the art », *IEEE Access*, vol. 8, p. 117782–117801, 2020.
[109] S. Wang, Y. Yuan, X. Wang, J. Li, R. Qin, et F.-Y. Wang, « An overview of smart contract: architecture, applications, and future trends », in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, p. 108–113.
[110] G. Wood, « Ethereum: A secure decentralised generalised transaction ledger », *Ethereum Proj. Yellow Pap.*, vol. 151, nº 2014, p. 1–32, 2014.
[111] I. Karamitsos, M. Papadaki, et N. B. A. Barghuthi, « Design of the Blockchain Smart Contract: A Use Case for Real Estate », *J. Inf. Secur.*, vol. 09, nº 03, p. 177, juin 2018.
[112] eiki, « What is EVM(Ethereum Virtual Machine)? », *Medium*, 07-juin-2019. .
[113] « Contract ABI Specification — Solidity 0.7.0 documentation ». [En ligne]. Disponible sur: https://docs.soliditylang.org/en/v0.7.0/abi-spec.html. [Consulté le: 24-juin-2022].

4 Annexe

4.1 La Blockchain Ethereum

4.1.1 Notion de compte Ethereum

Dans Ethereum, l'état est constitué d'objets appelés comptes. Chaque compte ayant une adresse de 20 octets et les transitions d'état étant des transferts directs de valeur et d'informations entre les comptes [106]. Un compte (EOA et comptes de contrats) Ethereum contient quatre champs, à savoir :

- Le **nonce**, un compteur utilisé pour s'assurer que chaque transaction ne peut être traitée qu'une seule fois (car incrémenté d'un pas après chaque transaction). Pour les EOA, il représente le nombre de transactions envoyées à partir de l'adresse. Alors que pour les comptes de contrats, il correspond au nombre de contrats créés par le compte.
- **Le solde actuel** en Ether du compte, il faut savoir que **1eth = 10e^{18} Wei**.
- **Le code de contrat du compte ou codeHash**, il correspond au hachage du code du compte dans la machine virtuelle d'Ethereum. Ainsi pour les EOA, c'est le hachage de la chaîne vide, alors que pour les comptes de contrats, c'est le nom du contrat qui est haché et stocké sous le nom codeHash.
- **Le stockage du compte** ou le **StorageRoot**, vide par défaut, est obtenu à partir du hachage des données du compte. Elle fait référence à la racine de Merkle.

L'éther est le principal crypto-carburant interne d'Ethereum, et est utilisé pour payer les frais de transaction. Un compte détenu en externe n'a pas de code, elle permet d'envoyer des transactions. Alors que l'on peut envoyer des messages à partir d'un compte de contrat, chaque fois que le compte de contrat reçoit un message, son code s'active, ce qui lui permet de lire et d'écrire dans le stockage interne et d'envoyer d'autres messages ou de créer des contrats à son tour.

4.2 Message et transaction dans Ethereum

Les messages dans Ethereum sont quelque peu similaires aux transactions dans bitcoin, mais avec trois différences importantes. Premièrement, un message Ethereum peut être créé soit par une entité externe, soit par un contrat, alors qu'une transaction bitcoin ne peut être créée qu'en externe.

Deuxièmement, il existe une option explicite pour que les messages Ethereum contiennent des données. Enfin, le destinataire d'un message Ethereum, s'il s'agit d'un compte de contrat, a la possibilité de renvoyer une réponse ; cela signifie que les messages Ethereum englobent également le concept de fonctions.

Une transaction permet de changer l'état global de la blockchain Ethereum. En ce sens, une transaction est une instruction signée avec des fonctions cryptographiques qui est générée par un compte externe, sérialisée, puis soumise à la Blockchain. Il existe deux types de transactions: les appels de message et les créations de contrat (c'est-à-dire les transactions qui créent de nouveaux contrats Ethereum) [107] [108]. Toutes les transactions (par abus de langage, un message est aussi considéré comme une transaction) sont composées des éléments suivants, quel que soit leur type :

- La propriété **From Account** : indique le compte qui est à l'origine de la transaction et représente un compte qui est prêt à envoyer du gas ou de l'Ether. Le compte peut être un compte externe ou un compte contractuel.
- Le **compte de destination** est un compte qui reçoit de l'Éther ou des avantages en lieu et place d'un échange. Pour les transactions liées au déploiement d'un contrat, le champ à est vide. Il peut être une propriété externe ou un compte de contrat.
- La propriété **value account** fait référence à la quantité d'Ether qui est transféré d'un compte à un autre.
- La propriété du **compte d'entrée** fait référence au contrat compilé et est utilisé pendant le déploiement du contrat dans EVM. Il est également utilisé pour stocker les données relatives aux appels ainsi que leurs paramètres. Une transaction typique dans Ethereum où une fonction de contrat est invoquée est illustrée ici. Dans la capture d'écran suivante, remarquez le champ de saisie contenant l'appel de fonction au contrat avec ses paramètres :
- La propriété du compte **blockHash** fait référence au hachage du bloc auquel cette transaction appartient.
- La propriété de compte **blockNumber** correspond au bloc auquel cette transaction appartient.
- La propriété du **compte gas** fait référence à la quantité de gaz fournie par l'expéditeur nécessaire pour exécuter cette transaction.

- La propriété du compte **gasPrice** fait référence au prix du gaz que l'expéditeur était prêt à payer en Wei (nous avons déjà appris à connaître le sur Wei dans la section Ether de ce chapitre). Le gas total est calculé en unités de **gaz * prix du gaz** appelé **Sartgas**. Afin d'éviter l'explosion exponentielle et les boucles infinies dans le code, chaque transaction doit fixer une limite au nombre d'étapes de calcul de l'exécution du code qu'elle peut engendrer, y compris le message initial et tous les messages supplémentaires qui sont engendrés au cours de l'exécution.
- La propriété **hash du compte** fait référence au hash de la transaction.
- La propriété du **compte nonce** fait référence au nombre de transactions effectuées par l'expéditeur avant la transaction actuelle c'est-à-dire avant la transaction en cours.
- La propriété de **compte transaction Index** fait référence au numéro de série des transactions en cours dans le bloc.
- La propriété de **compte value** fait référence à la quantité d'Éther transféré en Wei.
- Les propriétés de **compte v, r, et s** se rapportent aux signatures numériques de la transaction.

De plus, une transaction de création de contrat contient :

- **Init** : un tableau d'octets de taille illimitée spécifiant le code EVM pour la procédure d'initialisation du compte. Init est un fragment de code EVM ; il renvoie le corps, un deuxième fragment de code qui s'exécute à chaque fois que le compte reçoit un appel de message (soit via une transaction, soit en raison de l'exécution interne de code).

  Init n'est exécuté qu'une seule fois lors de la création du compte et est rejeté immédiatement après.

En revanche, une transaction d'appel de message contient :

- **Data (des données)** : un tableau d'octets de taille illimitée spécifiant les données d'entrée de l'appel de message. Une fonction annexe spécifie, qui mappe les transactions à l'expéditeur, et se produit via l'ECDSA de la courbe SECP-256k1, en utilisant le hachage de la transaction (à l'exception des trois derniers champs de signature) comme donnée à signer.

4.2.1 Exécution de transaction Ethereum

L'exécution d'une transaction est la partie la plus complexe du protocole Ethereum : elle définit la fonction de transition d'état. On suppose que toutes les transactions exécutées passent d'abord les tests initiaux de validité intrinsèque. Ceux-ci inclus :

- La transaction est un RLP (Recursive, Length, prefixe -en anglais) bien formé, sans octets de fin supplémentaires.
- La signature de la transaction est valide ;
- Le nonce de la transaction est valide (équivalent au nonce actuel du compte de l'expéditeur).
- La limite de gaz n'est pas inférieure au gaz intrinsèque, utilisé par la transaction.
- Le solde du compte expéditeur contient au moins le coût, requis dans le paiement initial.

Ainsi à la fin de l'exécution d'un contrat intelligent, nous avons en sortie un événement ou l'appel de message (qui contient des données et des valeurs).

4.3 La machine virtuelle Ethereum

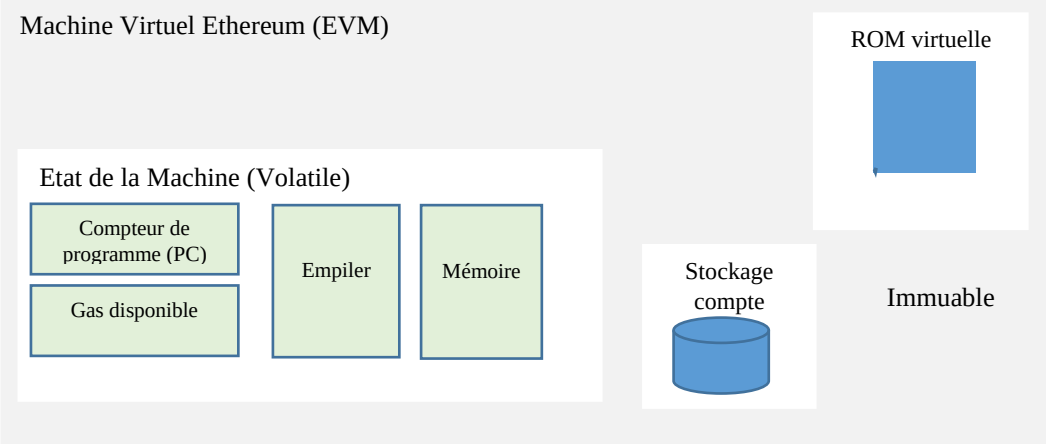


Figure 22: Description de la machine virtuelle Ethereum (EVM) et ces composants

Bien qu'Ethereum ait sa propre crypto-monnaie native (Ether) qui suit presque les mêmes règles intuitives. Il intègre également une fonction beaucoup plus puissante : les contrats intelligents. Pour cette caractéristique plus complexe, une analogie plus sophistiquée est requise. Au lieu de grand livre distribué, Ethereum est une machine à états distribuée.

L'EVM est une architecture simple basée sur une pile. La taille des mots de la machine (et donc la taille des éléments de pile) est de 256 bits. Ceci a été choisi pour faciliter le schéma de hachage Keccak256 et les calculs de courbe elliptique. Le modèle de mémoire est un simple tableau d'octets adressés par mot. La pile a une taille maximale de 1024. La machine a également un modèle de stockage indépendant ; c'est un concept similaire à la mémoire mais plutôt qu'un tableau d'octets, il s'agit d'un tableau de mots adressables par mot. Contrairement à la mémoire, qui est volatile, le stockage est non volatile et est maintenu dans le cadre de l'état du système. Tous les emplacements dans le stockage et la mémoire sont bien définis initialement comme zéro. La machine ne suit pas l'architecture standard de Von Neumann. Plutôt que de stocker le code du programme dans une mémoire ou un stockage généralement accessible, il est stocké séparément dans une ROM virtuelle interagissant uniquement via une instruction spécialisée. La machine peut avoir une exécution exceptionnelle, y compris les dépassements de capacité de pile et les instructions invalides. Comme l'exception de panne de gaz, ils ne laissent pas les changements d'état intacts. Au lieu de cela, la machine s'arrête immédiatement et signale le problème à l'agent d'exécution (soit le processeur de transaction, soit, de manière récursive, le générateur environnement d'exécution) qui le traitera séparément. Les redevances (libellées en gas) sont inculpées dans trois circonstances distinctes, toutes trois en tant que préalable à l'exécution d'une opération. La première et la plus courante est la commission intrinsèque au calcul de l'opération. Deuxièmement, le gaz peut être déduit afin de former le paiement pour un subordonné appel de message ou création de contrat ; cela fait partie du paiement pour CREATE, CREATE2, CALL et CALLCODE. Enfin, le gas peut être payé en raison d'une augmentation de l'utilisation de la mémoire. Au cours de l'exécution d'un compte, le total des frais d'utilisation de la mémoire à payer est proportionnel au plus petit multiple de 32 octets requis de telle sorte que tous les indices de mémoire (que ce soit pour la lecture ou l'écriture) sont inclus dans la plage. Ceci est payé sur une base juste-à-temps ; en tant que tel, référencer une zone de mémoire d'au moins 32 octets supérieure à toute mémoire précédemment indexée entraînera certainement des frais d'utilisation de la mémoire supplémentaires. En raison de ces frais, il est hautement improbable que les adresses ne dépassent jamais les limites de 32 bits. Cela dit, les implémentations doivent pouvoir gérer cette éventualité.

Les frais de stockage ont un comportement légèrement nuancé : pour inciter à minimiser l'utilisation du stockage (qui correspond directement à une base de données d'état plus importante sur tous les nœuds), les frais d'exécution pour une opération qui efface une entrée dans le stockage

ne sont pas seulement supprimés, le remboursement est accordé ; en fait, ce remboursement est effectivement payé d'avance puisque l'utilisation initiale d'un emplacement de stockage coûte beaucoup plus cher que l'utilisation normale. La compromission de plusieurs nœuds n'aura pas d'impact sur la sécurité globale du système.

4.3.1 Comment les transactions Ethereum sont minées ?

1. Un utilisateur écrit et signe une demande de transaction avec l'aide de sa clé privée du compte.
2. L'utilisateur diffuse la demande de transaction sur l'ensemble du réseau Ethereum à partir d'un nœud.
3. En entendant parler de la nouvelle de demande de transaction, chaque nœud du réseau Ethereum ajoute la demande à son mempool local, une liste de toutes les demandes de transaction dont il a entendu parler et qui n'ont pas encore été engagées dans un bloc de la blockchain.
4. En un moment donné, un nœud minier regroupe plusieurs dizaines ou centaines de demande de transaction dans un bloc potentiel, de manière à maximiser les frais de transaction qu'il gagne tout en restant sous la limite du gas de bloc. Le nœud de minage alors
    - Vérifie la validité de la demande de transaction (c'est-à-dire que personne n'essaie de transférer de l'Ether depuis un compte pour lequel il n'a pas produit de signature), puis exécute le code de la demande, en modifiant l'état de leur copie local de l'EVM. Le mineur attribue les frais de transaction pour chaque demande de transaction sur son propre compte.
    - Commence le processus de production du certificat de légitimité de la preuve de travail pour le bloc potentiel. Une fois que toutes les demandes de transaction dans le bloc ont été vérifiées et exécutées sur la copie de l'EVM local.

5. Finalement un mineur finira de produire un certificat pour un bloc qui comprend notre demande de transaction spécifique. Le mineur diffuse ensuite le bloc terminé, qui comprend le certificat et une somme de contrôle du nouvel état EVM revendiqué.

6. Les autres nœuds entendant parler de nouveau bloc. Ils vérifient le certificat, exécutent eux-mêmes toute les transactions sur le bloc (y compris la transaction initialement diffusée par notre utilisateur) et vérifient que la somme de contrôle de de leur nouvel état EVM après l'exécution de toute les transactions correspondent à la somme de contrôle de l'état revendiqué par le bloc mineur.

Ce n'est qu'alors que ces nœuds ajoutent ce bloc à la queue de leur Blockchain et acceptent le nouvel état EVM comme état canonique.

7. Chaque nœud supprime toutes les transactions du nouveau bloc de leur mempool local de demande de transaction non satisfaite.

8. Les nouveaux nœuds rejoignant le réseau téléchargent tous les blocs en séquence, y compris le bloc contenant notre transaction d'intérêt. Ils initialisent une copie EVM locale (qui commence comme une EVM à l'état vierge), puis passent par le processus d'exécution de chaque transaction dans chaque bloc au-dessus de leur copie EVM locale en vérifiant les sommes de contrôle d'état à chaque bloc en cours de route.

Nous allons aborder la notion de contrat intelligent et de machine virtuelle d'Ethereum.

4.4 Smart contract

4.4.1 Historique

En 1994, Nick Szabo a introduit les contrats intelligents, qui sont des programmes informatiques qui imitent les opérations décrites dans les contrats intelligents physiques/traditionnels.

Plus tard en 1996, Szabo a défini les objectifs suivants des contrats intelligents [109] : observabilité, vérifiabilité, confidentialité et force exécutoire. Avec ses attributs de lecture supplémentaires, Bitcoin et son langage de script montrent que la Blockchain est une plateforme appropriée pour la mise en œuvre de smart contract. Sur la blockchain, le contrat intelligent est immuable, il peut être facilement observé, vérifié et auto-appliqué et la confidentialité peut être assurée selon la méthode d'accès de la blockchain. Bien que son langage de script ne puisse que vérifier les transactions, bitcoin peut être considéré comme la première implémentation de contrats intelligents sur la blockchain.

En 2015, Vitalik Buterin a créé Ethereum, une plate-forme blockchain qui propose un système de paiement décentralisé et un langage complet Turing, qui permet le développement d'une grande variété de contrats intelligents sur une blockchain.

4.4.2 Définition

Un contrat intelligent est un ensemble de règles de réponse de scène et de logique de programme. En d'autre terme, il s'agit d'un code partagé, décentralisé et de confiance déployé sur une blockchain. Les signataires du contrat doivent convenir des détails du contrat, des conditions de résiliation, de la responsabilité en cas de rupture de contrat et de la source de données de

vérification (oracle), puis déployer en tant que contrat intelligent sur la blockchain pour présenter le signataire du contrat auto-exécuté [110]. L'ensemble du processus est indépendant de tout organisme central. Les contrats intelligents ont généralement les caractéristiques suivantes :

- Auto-exécutables : déclenchés par des transactions, sans avoir besoin d'interaction manuelle.
- Auto-exécutoire : une fois déclenchés, les contrats intelligents ne peuvent pas être empêchés de s'exécuter.
- Transparence : les contrats intelligents sont connus de chaque nœud du réseau Blockchain, car leur exactitude doit être vérifiée par la plupart des nœuds.
- Flexibilité : Ils peuvent s'adapter à différentes exigences de scénario. La popularité des contrats intelligents est également attestée par l'intérêt pour les langages et plates-formes de programmation de contrats intelligents.

Ainsi pour pouvoir effectuer une transaction, un utilisateur doit posséder au préalable un compte (ils sont de deux types) dans Ethereum.

4.4.3 Organisation d'un smart contract

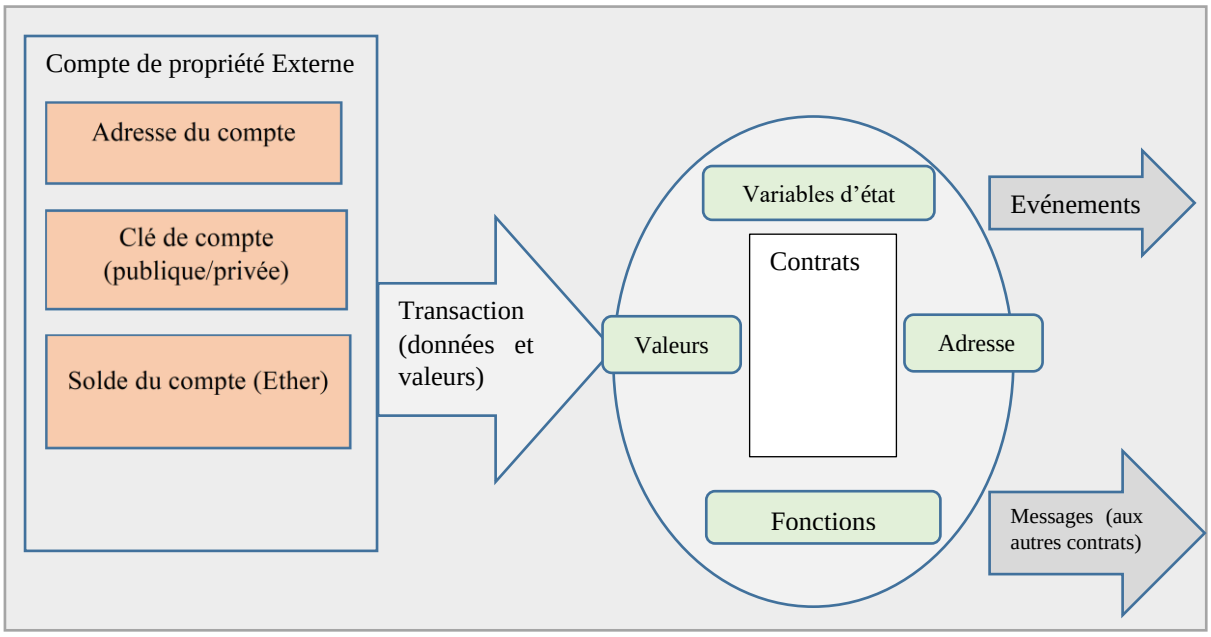


Figure 23: Activation d'un contrat intelligent

Considérez-le comme le cœur de la poussée de la blockchain Ethereum, un contrat est une collection de code (sa fonction) et de données (son état) qui réside à une adresse spécifique sur la

blockchain Ethereum. Cet ensemble d'instructions sur Turing nous permet de voir clairement la différence entre un message et une transaction. En fait, les comptes contractuels sont capables de transmettre des messages entre eux et d'effectuer des calculs presque complets de Turing. Les principaux composants d'un contrat intelligent sont les variables d'état, les fonctions, les modificateurs et les événements. Les contrats régissent le comportement des comptes dans l'état Ethereum[111] [112]. Solidity est un langage orienté objet de haut niveau pour la mise en œuvre de contrats intelligents. Par ailleurs il faut noter qu'il n'est pas le seul langage utilisé pour coder les contrats intelligents, nous avons Go, Python, C#, Java. Il s'agit d'un langage à typage statique qui prend en charge l'héritage, les bibliothèques et les types complexes définis par l'utilisateur, entre autres fonctionnalités.

4.4.4 Processus de compilation et de déploiement d'un contrat intelligent

La compilation est une étape fondamentale pour l'obtention du bytecode du contrat qui sera ensuite déployé sur le réseau Ethereum.

La compilation produira également l'interface binaire d'application (Application Binary Interface ABI en Anglais). Elle servira à appeler nos contrats via notre application.

Nous allons y revenir en détail sur l'ABI. Nous décrivons les différentes étapes nécessaires au déploiement de contrat.

- Le bytecode du contrat est généré après compilation du contrat
- Définir la limite de gas comme dans les autres transactions dans Ethereum (le déploiement nécessite beaucoup plus gas)
- Le plugin ou script de déploiement
- Accéder aux nœuds publics, soit en exécutant le vôtre, en se connectant à un nœud public, soit par le biais d'un API en utilisant les services de nœuds.

4.4.4.1 Bytecode EVM et la spécification ABI

- Ici, je vais clairement décomposer deux termes : bytecode et EVM bytecode. En informatique, le bytecode est un langage informatique qui est compilé à partir du code source et exécuté sur une machine virtuelle. Le bytecode n'est pas lisible par l'homme mais lisible par ordinateur. Une fois que vous comprenez ce qu'est le bytecode, il est facile de comprendre ce qu'est le « bytecode EVM ». Le byte code EVM est compilé à partir de Solidity et exécuté sur EVM [113] [113].

- ABI (Application Binary Interface) est une interface entre deux modules de programme ; souvent, entre un système d'exploitation et des programmes utilisateur. Dans Ethereum, ABI est une norme (ou un schéma), qui est défini comme une structure de données et des fonctions au format de tableau **JSON**, pour interagir avec un contrat intelligent à partir d'une application externe. Il permet l'interaction application-à-contrat et contrat-à-contrat. Lorsque les développeurs créent leur portefeuille ou leur DApp qui interagissent avec le contrat intelligent, l'application s'appelle ABI.

Le schéma suivant permet de décrire ce processus :

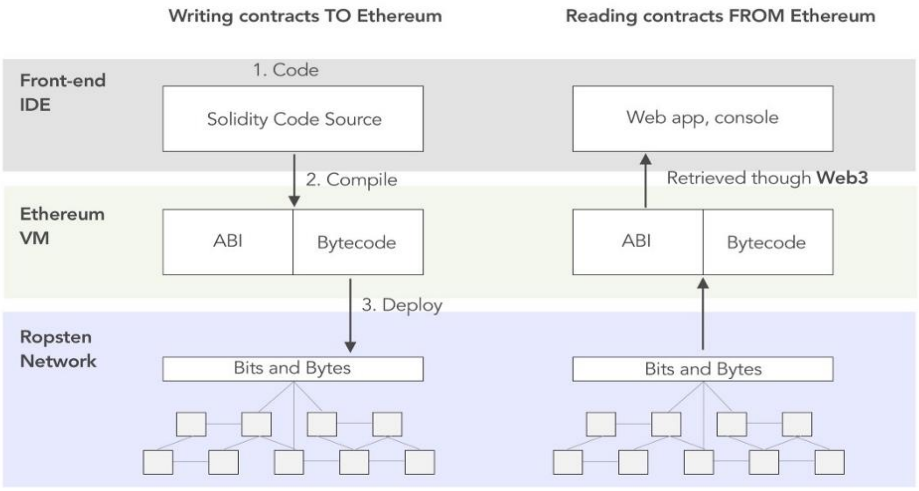


Figure 24: Processus de compilation et de déploiement d'un contrat intelligent