République algérienne démocratique et populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique
جامعة عين تموشنت بلحاج بوشعيب
Université d'Ain Temouchent - Belhadj Bouchaib
Faculté des Sciences et Technologie
Département de Mathématiques et Informatique

Projet de Fin d'Etudes

Pour l'obtention du diplôme de Master en : Informatique

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Cybersécurité et intelligence artificielle

Thème

Double Authentification (2FA) dans la Blockchain à l'aide de ZKP

Soutenu le : 23 / 06 /2024.

Par :

- M. BOUTERFES MOHAMED ACHRAF.
- M^{elle} DJEDOUI KHADIDJA.

Devant le jury composé de :

Dr MERAD BOUDIA Djalal.	MAA	UAT.B.B (Ain Temouchent)	Président
Dr BENOMAR Mohamed Lamine.	MCA	UAT.B.B (Ain Temouchent)	Examinateur
Dr MEDEDJEL Mansour.	MCA	UAT.B.B (Ain Temouchent)	Encadrant

Année Universitaire 2023/2024

DEDICACE

« À mon cher papa,

Ce travail est dédié à ta mémoire, à ton amour infini et à l'inspiration que tu as toujours été pour moi. Tu resteras à jamais dans mon cœur.

À ma tendre mère,

Ton soutien indéfectible, ta source de courage et ton amour inconditionnel ont toujours été mes piliers. Tes sacrifices et ta bienveillance ont rendu cette réussite possible.

À ma meilleure amie Fatima, Ta présence sans faille et ton soutien infini ont été ma source de réconfort. Cette réussite est aussi la tienne.

À ma famille et tous mes amis qui m'ont toujours encouragé, et à qui je souhaite plus de succès. À tous ceux que j'aime »

Khadidja

Je dédie de travail :

« À ma chère Mère,

Celle à qui mon cœur depuis sa naissance n'a pas pu éprouver qu'amour et reconnaissance, à celle qui a donné un sens à mon existence en m'offrant une éducation digne de confiance

À mon cher Papa,

Au symbole de sacrifice et courage mon père, pour son soutien moral depuis mon enfance.

À ma sœur,

À mes grands-parents,

À toute ma famille et mes amis. »

Achraf

À tous ceux qui ont croisé notre route et ont contribué à façonner notre parcours.

À vous tous, on dédie ce travail avec gratitude et humilité.

Merci pour tout.

REMERCIEMENTS

Ce travail n'aurait pas pu aboutir sans le soutien précieux de nombreuses personnes, que nous tenons à remercier sincèrement à travers ces quelques lignes.

Tout d'abord, nous exprimons notre gratitude envers Dieu, le tout Puissant et Miséricordieux, qui nous a donné la force et la patience d'accomplir ce modeste travail.

Nous adressons nos plus sincères remerciements à notre encadrant, le Dr. Mededjel Mansour, pour sa patience, sa disponibilité et ses conseils avisés qui ont grandement contribué à l'excellence de ce travail.

Nous sommes reconnaissants envers les membres du jury d'avoir accepté d'évaluer notre mémoire.

Nous remercions chaleureusement tous les enseignants du département de mathématiques et informatique, particulièrement ceux de notre spécialité "cybersécurité et intelligence artificielle", pour leur soutien continu et leurs précieuses contributions tout au long de notre parcours universitaire.

Enfin, nous tenons à remercier toutes les personnes qui ont contribuées de près ou de loin à la réalisation de ce mémoire.

Merci à tous.

TABLE DES MATIERES

CHAPITRE III : ZERO-KNOWLEDGE PROOF

CHPITRE IV : L'IMPLEMENTATION D'APPLICATION

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABREVIATIONS

AAA : Authentication, Authorization, and Accounting

ADSL : Asymmetric Digital Subscriber Line

API : Application Programming Interface

AS : Authentication Server

CBDC : Central Bank Digital Currency

CHAP : Challenge Handshake Authentication Protocol

DApps : Decentralized Applications

DHCP : Dynamic Host Configuration Protocol

DID : Decentralized Identifiers

DLT : Distributed Ledgers Technology

EAP : Extensible Authentication Protocol

GSC : Global Standards Collaboration

IA : intelligence artificielle

IBM : International Business Machines Corporation

IDE : Integrated Development Environment

IETF : Internet Engineering Task Force

IoT : l'Internet des Objets

IZKP : Interactive Zero-Knowledge Proof

JWT : JSON Web Tokens

KDC : Key Distribution Center

LDAP : Lightweight Directory Access Protocol

MFA : Authentification multifactorielle

MD5 : Message Digest 5

MS-CHAP : Microsoft Challenge Handshake Authentication Protocol

MS-CHAP-v1 : Microsoft Challenge Handshake Authentication Protocol-version 1

MS-CHAP-v2 : Microsoft Challenge Handshake Authentication Protocol-version 2

NAS : Network Access Server

NFT : Non-fungibles Tokens

NIP : Numéro d'Identification Personnel

NP : Non-deterministic Polynomial time

NP-Complet : Non-deterministic Polynomial-completeness

OAuth : Open Authorization

OSI : Open Systems Interconnection

P2P : peer to peer

PAE : Port Access Entity

PAP : Password Authentication Protocol

PCIe : Peripheral Component Interconnect Express

PKI : Public Key Infrastructure

PoET : Proof of Elapsed Time

PoS : Proof of stake

POW : Proof of Work

PPP : Point-to-Point Protocol

PPPoA : Point-to-Point Protocol over ATM

PPPoE : Point-to-Point Protocol over Ethernet

PXE : Preboot Execution Environment

RAM : Random Access Memory

RFC : Request for Comments

RTC : Réseau Téléphonique Commuté

SAML : Security Assertion Markup Language

SHA-256 : Secure Hash Algorithm 256-bit

SMS : Short Message Service

SNMP : Simple Network Management Protocol

SSO : Single Sign-On

SSL : Secure Sockets Layer

TACACS : Terminal Access Controller Access-Control System

TACACS+ : Terminal Access Controller Access-Control System Plus

TCP : Transmission Control Protocol

TBS : terabytes

TGS : Ticket Granting Server

TLS : Transport Layer Security

TPDU : Transport Protocol Data Unit

TPS : transactions par seconde

XTACACS : eXtended Terminal Access Controller Access-Control System

ZKP : zero knowledge proof

zk-SNARKs : Zero-Knowledge Succinct Non-Interactive ARgument of Knowledge

zk-STARKs : Zero-Knowledge Scalable Transparent Argument of Knowledge

INTRODUCTION GENERALE

INTRODUCTION GÉNÉRALE

Dans un monde où les craintes concernant la sécurité des données en ligne sont omniprésentes, les technologies de cryptage et de sécurisation des données sont devenues cruciales. Parmi celles-ci, la Blockchain se distingue comme une solution de confiance. En fonctionnant au sein d'un réseau décentralisé, la Blockchain offre un mécanisme transparent et sécurisé pour enregistrer toutes les transactions, sans nécessiter l'intervention d'intermédiaires. Cette innovation révolutionnaire a suscité un intérêt croissant dans divers domaines de la sécurité informatique.

Dans ce contexte, l'authentification occupe une place centrale en tant que première ligne de défense contre les cybermenaces. Son rôle essentiel est de confirmer l'identité des utilisateurs cherchant à accéder aux ressources et aux données sensibles d'une organisation. En garantissant que seules les personnes légitimes et autorisées peuvent franchir cette étape initiale de vérification, l'authentification contribue à réduire considérablement les risques d'accès non autorisés et de violations de la sécurité.

Parallèlement, la découverte et la divulgation des vulnérabilités représentent également des défis majeurs pour la protection des systèmes et des utilisateurs. Dans ce contexte, l'émergence des technologies de preuve à divulgation nulle de connaissance (Zero-knowledge proof - ZKP) ouvre de nouvelles perspectives. Ces technologies permettent de prouver la validité d'une déclaration sans révéler d'autres informations que cette validité, offrant ainsi une approche plus sécurisée de la divulgation des vulnérabilités.

Dans ce mémoire, nous explorerons en détail l'importance, les méthodes et les technologies de l'authentification dans le contexte complexe de la sécurité informatique, en mettant particulièrement l'accent sur l'utilisation du Zero-knowledge proof (ZKP) dans la Blockchain. Chaque chapitre examinera en profondeur un aspect spécifique de ce sujet :

- Le premier chapitre se concentrera sur les principes fondamentaux de la Blockchain et son rôle dans la sécurisation des transactions numériques.

- Le deuxième chapitre explorera les mécanismes traditionnels d'authentification et leur pertinence dans un environnement blockchain.

- Le troisième chapitre examinera les avantages et les applications du Zero-knowledge proof (ZKP) dans la Blockchain.

Enfin, le quatrième chapitre constituera une implémentation pratique d'une application de Double Authentification (2FA) dans la Blockchain à l'aide de ZKP. Nous examinerons comment cette approche innovante renforce la sécurité des transactions en ajoutant une couche supplémentaire d'authentification, tout en préservant la confidentialité des données des utilisateurs. Des recommandations et des lignes directrices seront également fournies pour aider à intégrer cette solution dans d'autres applications blockchain, ouvrant ainsi la voie à des transactions numériques plus sûres et plus fiables.

CHAPITRE I :

LA TECHNOLOGIE BLOCKCHAIN

CHAPITRE I : LA TECHNOLOGIE BLOCKCHAIN

1. Introduction

La sécurité des données en ligne est devenue une préoccupation majeure dans notre ère numérique. Chaque jour, les médias relatent des incidents de piratage et d'intrusion dans les systèmes en ligne, mettant en évidence le besoin croissant de solutions de cryptage et de sécurisation des données. Dans ce contexte, la blockchain émerge comme une technologie révolutionnaire et prometteuse.

La blockchain, bien que sa définition soit complexe, peut être considérée comme une base de données distribuée, décentralisée et partagée permettant le transfert et le stockage de valeurs ou d'actifs via Internet. Sa force principale réside dans sa capacité à éliminer la nécessité de faire confiance à un tiers, grâce à un mode de gouvernance décentralisé et un fonctionnement de pair à pair.

Dans ce chapitre, nous aborderons les concepts généraux de la blockchain, y compris ses principes de base et son fonctionnement global. Cette exploration jettera les bases nécessaires à la compréhension de son rôle crucial dans l'évolution des transactions numériques et de son potentiel d'application dans divers secteurs d'activité.

2. Historique

Les idées fondamentales derrière la technologie blockchain ont émergé dans les années 1980 et 1990. En 1989, Leslie Lamport a développé le protocole Paxos, décrivant un modèle de consensus pour parvenir à un accord dans un réseau d'ordinateurs où les défaillances individuelles peuvent survenir. En 1991, une chaîne d'informations signées a été utilisée comme registre électronique pour signer numériquement des documents, fournissant une preuve facile que les documents n'avaient pas été modifiés. Ces concepts ont posé les bases de la blockchain. [1]

Cependant, le véritable essor de la technologie blockchain est survenu avec l'introduction du Bitcoin en 2008 par une personne ou un groupe sous le pseudonyme de Satoshi Nakamoto. Le Bitcoin a été la première application concrète de la blockchain, utilisée comme registre public pour enregistrer toutes les transactions de manière sécurisée et transparente, sans nécessiter l'intervention d'une autorité centrale. Depuis lors, la blockchain a évolué pour être utilisée dans une variété d'applications au-delà des crypto-monnaies, offrant la promesse de sécurité, de transparence et d'immuabilité des données dans divers domaines. Aujourd'hui, la technologie blockchain est largement étudiée et adoptée, ouvrant la voie à de nouvelles formes d'innovation et de transformation numérique. [2]

3. Définition de la Blockchain

La Blockchain est fondamentalement une technologie de stockage et de transmission d'informations sécurisées, à l'image d'une base de données distribuée, en y intégrant en plus une protection cryptographique des données et en permettant la conservation de l'historique de tous les échanges effectués entre ses participants. Echange de valeurs, transfert de propriété, ou encore notarisation des transactions qui se réalisent grâce à une chaîne de blocs contenant les données, d'où le terme « block » - « chain ». Mais à la différence d'une base de données classique, la Blockchain

introduit un nouveau type de gouvernance décentralisée, intégrée et gérée par la technologie, sans intermédiaire qui ne requiert pas la présence d'une tierce autorité de contrôle. [3]

En d'autres termes, une Blockchain, est une technologie de stockage et de transmission d'informations sans organe de contrôle. Techniquement, il s'agit d'une base de données distribuée dont les informations envoyées par les utilisateurs et les liens internes à la base sont vérifiés et groupés à intervalles de temps réguliers en blocs, l'ensemble étant sécurisé par cryptographie, et formant ainsi une chaîne [4].

En Outre, nous pouvons définir la Blockchain comme un grand livre décentralisé et distribuéé composé de nombreux pairs (nœuds), ou chaque pair a les mêmes enregistrements de données appelés transactions qui sont structurées dans des blocs formant ainsi une chaîne [5] , comme vous pouvez le constater dans la figure 1.

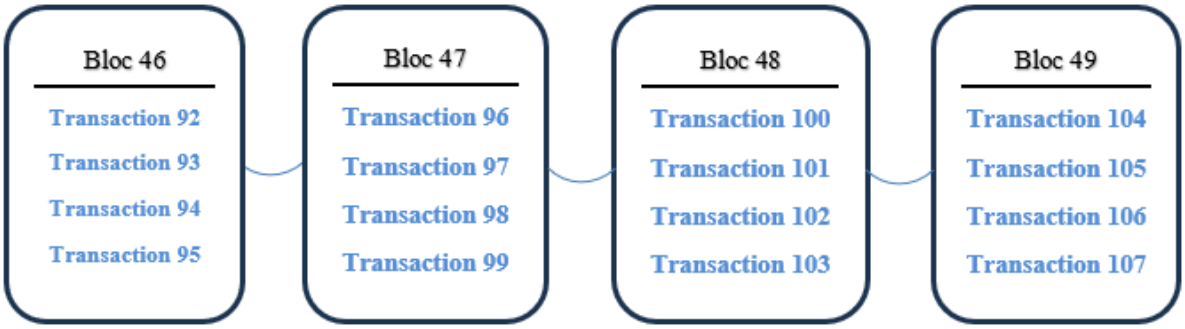

Figure 1– Aperçu d'une Blockchain. *[4]*

La blockchain permet donc, de lier un ensemble de règles et de procédures à des transactions spécifiques afin de normaliser les activités du processus.

Ses caractéristiques, en particulier sa transparence et son approche de consensus partagé, offre aux utilisateurs la possibilité de rendre les transactions irrévocables (immuabilité), ce qui augmente la fiabilité des documents tout en simplifiant les processus de soutien et de contrôle.

En effet, la technologie blockchain permet le transfert d'actifs en éliminant les intermédiaires tiers. Bien que ses premières applications se soient concentrées sur les marchés financiers, ses implications dépassent largement ce secteur. En exploitant les réseaux pairs à pair et en supprimant les intermédiaires, la blockchain offre de nouvelles opportunités de digitalisation, réduit les coûts opérationnels et accélère le règlement des transactions.

Au-delà de la finance, la blockchain a le potentiel de transformer diverses industries, y compris l'audit, la comptabilité et bien d'autres. De nombreuses entreprises investissent dans la recherche pour réinventer leurs pratiques commerciales, tirant parti des avantages offerts par cette technologie innovante. [6]

4. L'architecture de la Blockchain

L'architecture d'une blockchain repose sur un réseau distribué de type pair-à-pair (P2P). La notion de pair-à-pair (Peer-to-Peer ou P2P), définit un réseau informatique où les ordinateurs communiquent

directement sans l'intervention d'un tiers. Sur Internet, cette communication directe entre ordinateurs en mode P2P élimine la nécessité d'un intermédiaire, comme indiqué dans la figure 2.

La blockchain fonctionne au sein de ce réseau distribué de serveurs, également appelés nœuds, qui cherchent à atteindre un consensus sur l'état actuel de la blockchain. Chaque nœud détient une copie de la blockchain, agissant comme un grand livre de transactions sécurisé, similaire à un registre public. Cette structure confère au système une sécurité élevée et une transparence accrue en tant que système décentralisé.

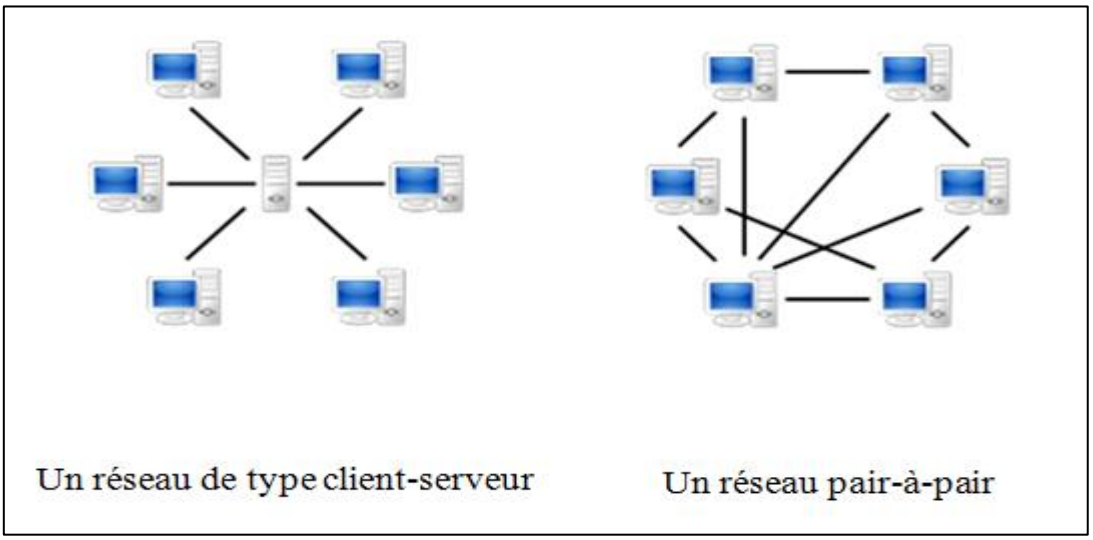

Figure 2- Architecture Client/serveur vs Architecture P2P. [7]

5. Les fondements de la Blockchain

La blockchain est fondamentalement une base de données distribuée publique qui maintient un grand livre chiffré. Contrairement à une base de données classique stockée sur un serveur central, chaque participant dans une blockchain détient une copie de chaque enregistrement. Un bloc est une structure de données composée des enregistrements les plus récents, et le bloc initial d'une blockchain est appelé "Genesis". [8]

Chaque bloc suivant intègre le hachage de son prédécesseur, rendant toute altération immédiatement détectable grâce à la nature non inversible et résistante aux collisions du hachage, comme illustré dans la figure 3.

La blockchain conserve un grand livre horodaté, sécurisé, chronologique et immuable, avec les transactions comme éléments de base. Chaque participant ou dispositif est appelé un nœud, et les mineurs sont des nœuds spécifiques chargés de vérifier les blocs. Les règles régissant les opérations dans une blockchain sont définies par le consensus. Chaque bloc est visible pour tous les participants, mais aucune modification ne peut être apportée sans validation par au moins 51 % des pairs, ce qui constitue la preuve de consensus. La preuve de travail nécessite également que les nouveaux nœuds résolvent des problèmes mathématiques, tandis que la preuve d'enjeu exige que chaque nœud mette quelque chose en jeu, assurant ainsi la sécurité et la confidentialité de la blockchain contre toute ingérence malveillante.

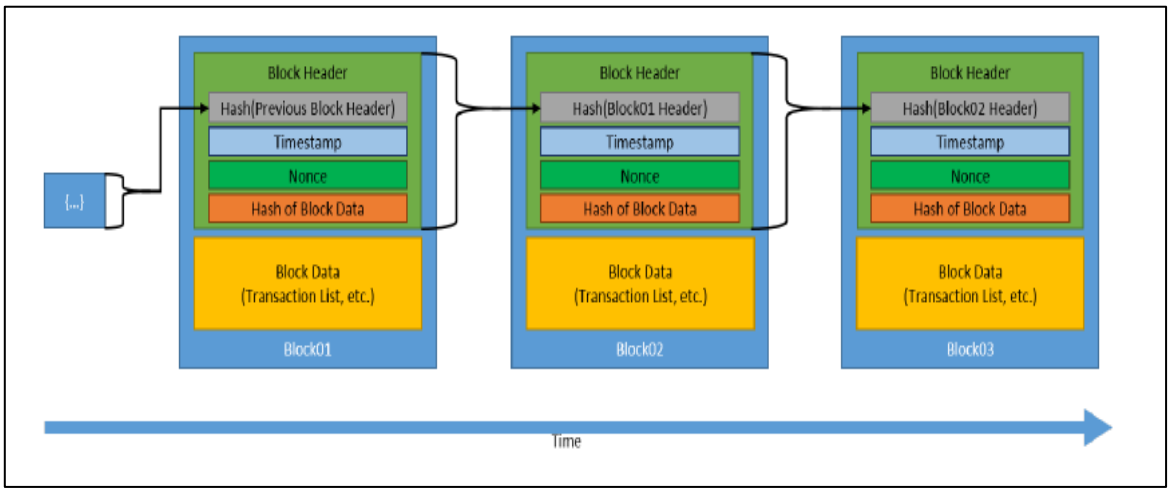

Figure 3– Structure de la blockchain. [1]

6. Fonctionnement de la Blockchain

Afin de mieux appréhender en quoi la Blockchain peut apparaître comme révolutionnaire, il est nécessaire de rappeler le contexte dans lequel elle a été développée.

À l'origine, les utilisateurs recherchaient un système efficace, rentable et sécurisé pour effectuer et enregistrer des transactions financières [9]. Bien que le système bancaire fonctionne assez bien pour la plupart des transactions, il souffre toujours de faiblesses inhérentes au modèle basé sur la confiance et réputé trop cher. [2]

La blockchain repose principalement sur la technologie des registres distribués (DLT, Distributed Ledgers Technology). Il s'agit d'un journal de transactions enregistré et synchronisé sur un réseau d'ordinateurs, diffusé à travers un réseau P2P. Un algorithme de consensus est intégré pour synchroniser et authentifier l'état du registre, formant ainsi une chaîne de blocs, communément appelée "blockchain". Cette structure distribuée confère à la blockchain une sécurité supplémentaire par rapport aux architectures classiques "Client-Serveur". [10]

Les blocs de la chaîne sont ordonnés selon des numéros générés par une fonction de hachage, assurant l'ordre immuable des blocs. La blockchain utilise un algorithme de consensus pour parvenir à un accord sur l'état historique des transactions sans entité centralisée, créant ainsi une "machine digne de confiance". L'introduction des smart contrats permet d'appliquer une logique métier complexe à la blockchain, automatisant des calculs et des processus de contrôle.[1]

Il est également important de noter deux types essentiels de blockchains : les publiques (public ou permissionless) et les privées (private ou permissioned). Les blockchains privées restreignent l'accès au réseau peer-to-peer à certains participants, offrant un contrôle plus étroit sur l'accès, la vérification et l'ajout de transactions au registre.

6.1. Principe Général d'une Blockchain :

Pour une première approche du fonctionnement des blockchain, le plus facile est de raisonner avec une blockchain purement monétaire. On peut prendre l'exemple de Bitcoin, ou d'une blockchain avec

des jetons « simples », en commençant par la création d'une transaction peut être décrit en quelques étapes, comme présenté dans la figure 4. :

1. Un compte (ou portefeuille, portemonnaie, wallet) doit être crée pour qu'un utilisateur de blockchain puisse envoyer ou recevoir des crypto-monnaie. A utilise son portefeuille et effectue une transaction vers B. Cette transaction est diffusée sur le réseau.

2. A la réception de la transaction, chaque mineur authentifie la transaction à l'aide de la clé publique de A. Cette transaction avec d'autres transactions récentes sont regroupées en bloc, et chaque transaction sera vérifiée et validée par les mineurs. Lors de la vérification de la transaction, l'historique des transactions de A est remonté pour vérifier que l'argent qu'il a reçu précédemment n'a pas été réutilisé depuis. On a vérifié en fait tout simplement qu'il n'essaye pas de dépenser deux fois l'argent qu'il a reçu.

3. Une fois les vérifications effectuées, le bloc dans lequel se trouve la transaction entre A et B est validé par les mineurs, selon des techniques de consensus qui dépendent du type de blockchain, et qui permettent d'atteindre le consensus distribué, c'est-`a-dire le consensus des nœuds sur l'état du réseau. [12]

Pour cela, les mineurs doivent lancer un procédé cryptographique : le calcul du hash du bloc. Chaque bloc possède un identifiant qui prend la forme d'une hache permettant de relier les blocs les uns aux autres. Cet hash est toujours le résultat du hachage du bloc précédent.

4. Quand le bloc est validé, il est daté et ajouté à la chaîne de blocs à laquelle tous les utilisateurs ont accès.

5. Enfin, "B" reçoit la transaction de A.

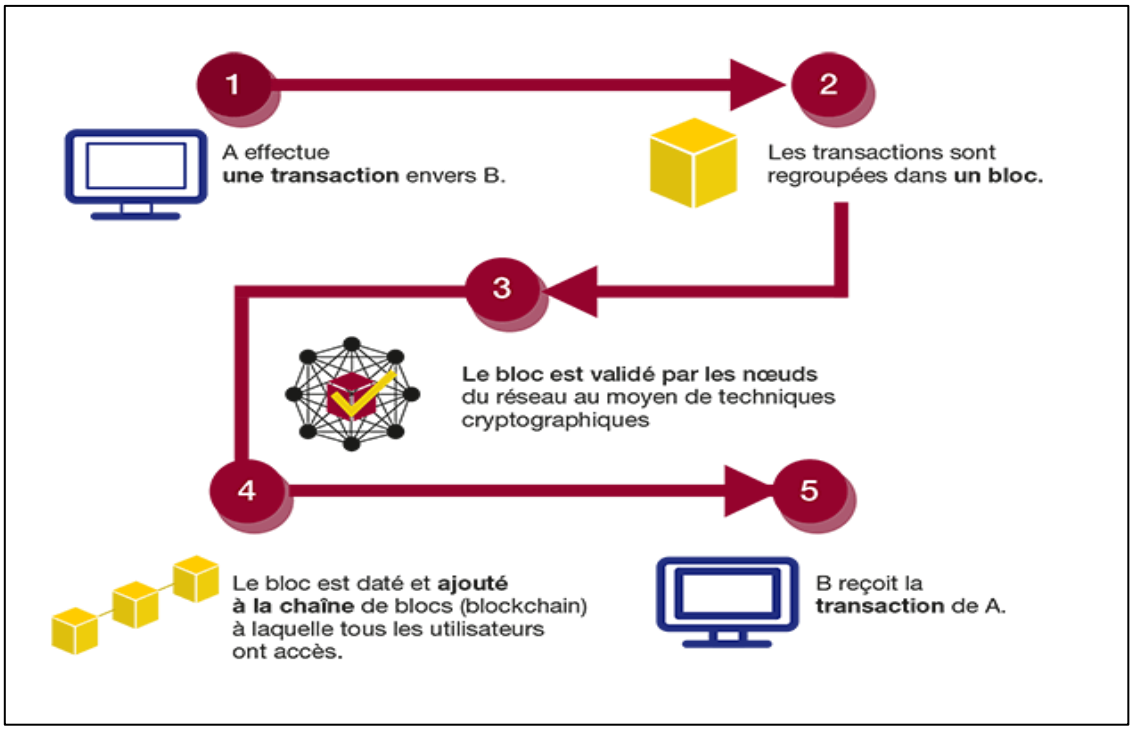

Figure 4– Principe Général d'une Blockchain [13]

7. Composants de Blockchain

La technologie de blockchain se base sur plusieurs concepts comme montré dans l'architecture de la figure 5:

❖ Compte ou porte-monnaie (wallet) :

Le porte-monnaie est un logiciel pour la sauvegarde de la clé privé et clé publique d'un utilisateur de blockchain, ce logiciel peut être une application sur le web, téléphone mobile ou ordinateur. Il permet à cet utilisateur de contrôler son compte de crypto-monnaie, voir son montant et exécuter des transactions.

- ***Transaction :***

Le Concept des transactions repose en réalité sur la cryptographie asymétrique. Lorsque quelqu'un se lance dans le processus de création d'une transaction, il génère, à l'aide de son porte-monnaie, une clé publique et une clé privée. La clé privée n'est transmise à personne et la clé publique est, quant à elle, disponible pour tout le monde. Les deux clés (privée et publique) permettent de garantir l'intégrité des données transmises en chiffrant les données envoyées, ainsi que l'authentification de l'origine de la transaction.

- ***Bloc :***

Un bloc est un enregistrement dans la blockchain, qui contient et confirme plusieurs données ou transactions en attente. Toutes les 10 minutes en moyenne (dans le cas de Bitcoin), un nouveau bloc contenant des transactions est ajouté à la chaine de blocs par le minage. [14]

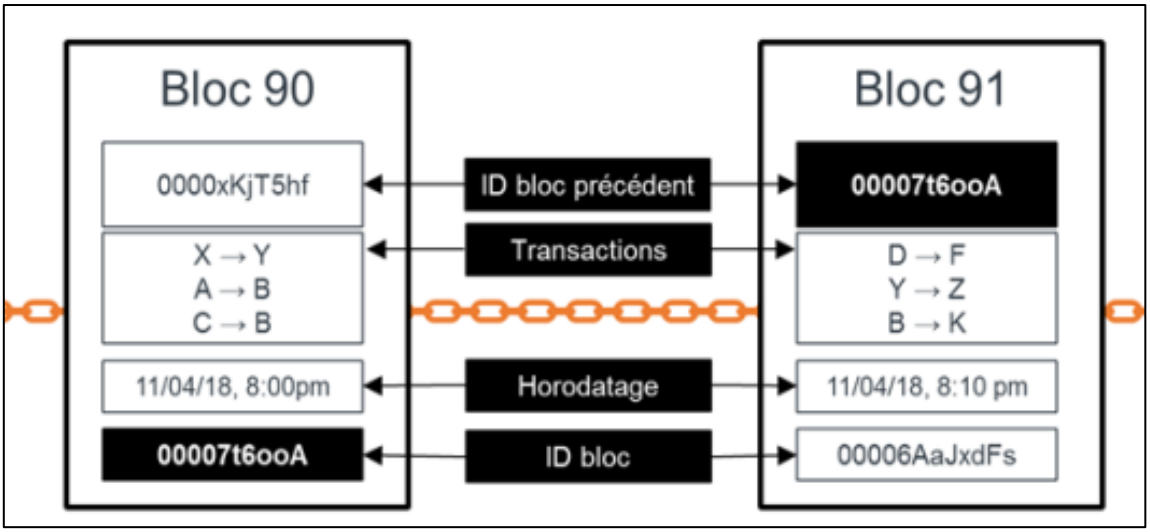

Figure 5 - Architecture interne de blockchain.[13]

Chaque bloc est constitué de plusieurs champs :

- L'indice de bloc.
- Hash du bloc précèdent : champ contient l'empreinte de bloc précédant (bloc d'indice numéro 91 dans notre exemple).
- Transaction : la partie qui contient la liste des transactions.
- Horodatage : temps de la création de bloc.

- Hash : l'identifiant du bloc actuel.

➢ ***Minage (mining) :***

Le minage est un processus très important dans une blockchain [15]. C'est le processus qui permet la validation des blocks des transactions. Cette opération, très coûteuse en puissance de calcul informatique, est motivée par l'obtention d'une récompense par le mineur gagnant.

➢ ***Le consensus:***

Un consensus joue un rôle important dans le contexte de la Blockchain. Le but est que tous les participants s'accordent sur un seul état de la blockchain. Puisqu'il n'y a pas de centralisation pour décider quels nouveaux blocs sont valides, chaque nœud doit décider s'il accepte un nouveau bloc reçu ou non. Étant donné que tous les nœuds du réseau ont leur propre copie des données complètes de la blockchain et ils utilisent la signature numérique pour vérifier l'authenticité de toutes les transactions. [4]

Le mécanisme de consensus a les objectifs suivants :

✓ S'assurer d'avoir un bloc valide dans la chaine de blocs.
✓ Parvenir à un accord équitable pour toutes les parties concernées
✓ Il faut s'assurer qu'il n'y aura pas d'adversaires puissants pour réussir à bifurquer la chaîne.
✓ Rendre le réseau plus solide face aux divers types d'attaques.

Les algorithmes de consensus constituent une partie très importante des blockchains. Il existe divers types de consensus, et ces modèles définissent quel nœud est responsable du stockage du bloc suivant et comment la validation du nouveau bloc ajouté doit être effectuée.

Dans la section suivante, nous illustrons les différents algorithmes de consensus les plus utilises.

8. Les algorithmes de Consensus

8.1. Proof of work (PoW):

Dans le modèle de preuve de travail, un utilisateur publie le bloc suivant en étant le premier à résoudre un problème intensif en calcul. La solution à ce problème est la "preuve" qu'ils ont effectué le travail. Le problème est conçu de telle sorte qu'il est difficile de le résoudre, mais il est facile de vérifier qu'une solution est valide. Cela permet à tous les autres nœuds de valider facilement tous les blocs suivants proposés, et tout bloc proposé qui ne satisfait pas la solution serait rejeté. Cet algorithme est utilisé par Bitcoin et beaucoup d'autres.[16]

❖ **Objectif :**

Fournir un obstacle à la publication de blocs sous la forme d'un casse-tête Difficile à résoudre pour permettre les transactions entre des participants non fiables.

❖ **Avantages :**

- Difficulté d'effectuer un déni de service en inondant le réseau de blocs défectueux.

- Ouvert à toute personne disposant de ressources.

❖ **Inconvénients**

- Calcul intensif, consommation d'énergie, course aux armements matériels.
- Potentiel d'attaque de 51 % en obtenant suffisamment de puissance de calcul.

8.2. Proof of stake (PoS):

Ce modèle repose sur l'idée que les utilisateurs investis dans le système ont plus d'intérêt à sa réussite et moins de motivation à le renverser. Il est utilisé par Ethereum, Casper, et Krypton. [17]

❖ **Objectif :**

Rendre le calcul des blocs moins intensif tout en autorisant les transactions entre des participants peu fiables

❖ **Avantages :**

- Moins intensif en calcul que PoW.
- Ouvert à tous ceux qui souhaitent miser des crypto-monnaies.

❖ **Désavantages :**

- Rien n'empêche la constitution d'un pool d'acteurs pour créer un pouvoir centralise.
- Potentiel d'attaque de 51% en obtenant suffisamment de puissance financière.

8.3. Delegated Proof of stake (dPoS):

Le Proof-of-Stake delegué (dPoS) est une version du mécanisme de consensus Proof-of-Stake (PoS) qui intègre une "couche de démocratie" dans la validation des blocs, favorisant ainsi une plus grande équité et transparence. Dans ce système, les détenteurs de jetons peuvent voter pour des délégués chargés de valider les transactions en leur nom, ajoutant ainsi une dimension démocratique au processus.Il est utilisé par des projets importants dans l'écosystème des cryptomonnaies et des blockchains comme : Bitshares, Steem, Cardano. [18]

❖ **Objectif :**

Faciliter un modèle de consensus dynamique avec une démocratie directe où les participants votent (via des messages cryptographiques signés) pour choisir ou révoquer les droits des délégués chargés de valider et sécuriser la Blockchain.

❖ **Avantages :**

- Transparence et responsabilité grâce à l'élection des délégués par les détenteurs de tokens.
- Efficacité accrue avec un nombre limité de délégués pour un traitement plus rapide des transactions.

- Participation plus large des petits détenteurs grâce à leur implication dans le choix des délégués.

❖ **Inconvénients :**

- Risque de collusion entre délégués, compromettant ainsi l'intégrité du réseau.
- Complexité accrue du système due aux processus électoraux et à la gestion des délégués.
- Risque de centralisation si quelques grands détenteurs de tokens contrôlent la majorité des votes.

8.4. Proof of Authority (PoA):

Le modèle de consensus de preuve d'autorité (également appelée preuve d'identité) repose sur la confiance partielle des nœuds de publication via leur lien connu avec les identités du monde réel. Les nœuds de publication doivent avoir leur identité prouvée et vérifiable dans le réseau de la blockchain (par exemple, les documents d'identification qui ont été vérifiés et notariés et inclus dans la blockchain). L'idée est que le nœud de publication mise son identité/réputation pour publier de nouveaux blocs. Les utilisateurs du réseau Blockchain affectent directement la réputation d'un nœud de publication en fonction du comportement du nœud de publication. Les nœuds de publication peuvent perdre de la réputation en agissant d'une manière avec laquelle les utilisateurs du réseau blockchain ne sont pas d'accord, tout comme ils peuvent gagner en réputation en agissant d'une manière avec laquelle les utilisateurs du réseau blockchain sont d'accord, utilisé par Ethereum Kovan testnet, PoA Chain, divers systèmes autorisés utilisant Parity. [1]

❖ **Objectif :**

Créer un processus de consensus centralisé pour minimiser le taux de confirmation de blocs.

❖ **Avantages :**

- Temps de confirmation rapide.
- Permet des taux de production de blocs dynamiques.
- Peut être utilisé dans les chaînes latérales pour les réseaux de chaînes de blocs qui utilisent un autre modèle de consensus.

❖ **Inconvénients :**

- Se fonde sur l'hypothèse que le nœud de validation actuel n'a pas été compromis.
- Conduit à des points de défaillance centralisés.
- La réputation d'un nœud donné est sujette à un risque de queue (tail risk), car elle peut être compromise à tout moment.

8.5. Proof of Elapsed Time (PoET):

Dans le cadre du modèle de consensus PoET (Proof of Elapsed Time), chaque nœud de publication attend un délai déterminé par une source de temps matérielle sécurisée au sein de son système informatique. Cette source génère un temps d'attente aléatoire, renvoyé ensuite au logiciel du nœud. Après avoir reçu cette valeur, le nœud devient inactif pour la durée spécifiée. Une fois le délai écoulé, le nœud se réactive, crée un nouveau bloc, et le publie sur le réseau de la chaîne de blocs, signalant

ainsi aux autres nœuds l'existence du bloc. Ce processus se répète pour chaque nœud, assurant un consensus fiable et sécurisé. Cette méthode est notamment utilisée par Hyperledger Sawtooth.[19]

❖ **Objectif :**

Permettre un modèle de consensus plus économique pour les réseaux de chaines de blocs, au détriment de garanties de s´sécurité plus approfondies associées à PoW.

❖ **Avantage :**

- Moins coûteux en calcul que PoW.

❖ **Désavantages :**

- Configuration matérielle requise pour gagner du temps.
- Suppose que l'horloge matérielle utilisée pour d´dériver l'heure n'est pas compromise.
- Compte tenu des limites de vitesse de latence tardive, la synchronicité en temps réel est essentiellement impossible dans les systèmes distribués.

8.6. Round Robin :

Le Round Robin est un modèle de consensus adopté par certains réseaux blockchain autorisés. Dans ce modèle, les nœuds produisent des blocs tour à tour. Il repose sur une longue tradition dans l'architecture des systèmes distribués. Pour éviter les interruptions dues à l'indisponibilité d'un nœud pour publier un bloc à son tour, une limite de temps peut être instaurée pour permettre aux nœuds disponibles de prendre le relais. Ainsi, aucun nœud ne monopolise la création des blocs. Ce modèle se distingue par sa simplicité, son absence d'énigmes cryptographiques et sa faible consommation énergétique. [1]

❖ **Objectif :**

Assurer un mécanisme de publication de blocs au sein des nœuds de publication agréés.

❖ **Avantages :**

- Faible demande en puissance de calcul.
- Facile à comprendre.

❖ **Désavantage :**

- Requiert une confiance élevée entre les nœuds de publication

9. Types de blockchains

Les Blockchains peuvent être catégorisées en fonction de leur accessibilité, déterminant s'ils sont ouverts sans restriction ou soumis à l'approbation d'un tiers. On distingue ainsi les blockchains ouvertes (permissionless) et fermées (permissioned), également appelées publiques ou privées. [13]

Globalement, il existe quatre principaux types de réseaux blockchain, comme indiqué dans la figure6:

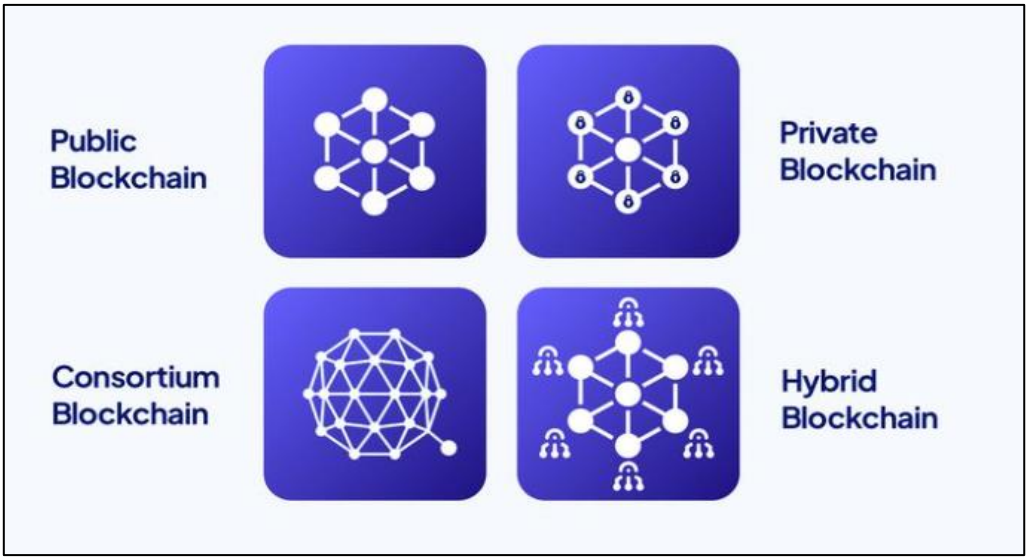

Figure 6- Les différents types de Blockchain.

9.1. Blockchain publique :

La blockchain publique est contrôlée par l'ensemble des nœuds du réseau en mode peer-to-peer. Elle présente une accessibilité totale, sans barrière d'entrée ni nécessité d'autorisation pour effectuer des transactions. Tous les participants sont sur un pied d'égalité, contribuant à un réseau distribué de grande envergure, ouvert à tous les niveaux, et avec un code source accessible et maintenu par la communauté, à l'instar de Bitcoin.

9.2. Blockchain privée :

La blockchain privée tourne sur un réseau privé sur lequel le gérant peut modifier le protocole quand il le souhaite.

La blockchain privée fonctionne dans un environnement restrictif comme un réseau fermé, ou qui est sous le contrôle d'une seule entité. Ce sont plus petites que les autres types, leur accès est complétement contrôlé. Personne ne peut y participer sans y être autorisé.

Bien qu'il fonctionne comme un réseau blockchain public dans le sens où il utilise des connexions pair-à-pair et la décentralisation, ce type de blockchain est à une échelle beaucoup plus petite. Au lieu que n'importe qui puisse s'y joindre et fournir de la puissance de calcul, les blockchains privées sont généralement exploitées sur un petit réseau au sein d'une entreprise ou d'une organisation. Elles sont également appelées blockchains à autorisation ou blockchains d'entreprise.

"La technologie blockchain utilisée pour soutenir différents projets peut être la même, mais les utilisateurs finaux ciblés peuvent varier", a expliqué Espae Hong, responsable de l'Institut de recherche sur la blockchain de la CBDC. Les blockchains publiques visent le grand public en tant qu'utilisateurs finaux, tandis que les blockchains privées sont destinées aux utilisateurs sur invitation seulement et à leurs réseaux, a-t-elle ajouté.

9.3. Blockchain hybride :

Ce type de blockchain est une fusion entre blockchains publiques et privées. C'est une tentative d'obtenir le meilleur des deux mondes. C'est une technologie qui combine des éléments de blockchain privée et publique. Elle permet aux organisations de mettre en place un système privé, basé sur des autorisations, parallèlement à un système public sans autorisation, ce qui leur permet de contrôler qui peut accéder à des données spécifiques stockées dans la blockchain, et quelles données seront ouvertes au public.

Dans ces blockchains, la participation au réseau est privée. C'est-à-dire, l'accès aux ressources du réseau est contrôlé par une ou plusieurs entités. Toutefois, le registre est accessible au public. Cela signifie que n'importe qui peut explorer bloc par bloc tout ce qui se passe sur la blockchain. [20]

En général, les transactions et les enregistrements d'une blockchain hybride ne sont pas rendus publics mais peuvent être vérifiés si nécessaire, par exemple en autorisant l'accès par un contrat intelligent.[21]

9.4. Blockchain consortium :

Également appelée blockchain fédérée, est similaire à une blockchain hybride en ce sens qu'elle possède des caractéristiques de blockchain privée et publique. Mais elle est différente dans la mesure où plusieurs membres de l'organisation collaborent sur un réseau décentralisé. Essentiellement, une blockchain de consortium est une blockchain privée dont l'accès est limité à un groupe particulier, ce qui élimine les risques liés au contrôle du réseau par une seule entité sur une blockchain privée.[22]

10.Evolution de la blockchain

Depuis la naissance du Bitcoin, la Blockchain a constamment progressé et évolué. À ce jour, cette technologie a connu quatre évolutions majeures :

10.1. Blockchain 1.0: « Consensus décentralisé »

C'est la première génération de blockchain, elle est issue du concept de la Technologie des Registres Distribués (DLT).[23]

Le registre distribué, partagé consensuellement entre plusieurs participants, a été introduit pour éliminer les scénarios de double dépense. Cette première phase a été largement associée à la cryptomonnaie, en particulier Bitcoin, qui a joué un rôle central en devenant la "monnaie d'internet ». Le processus utilise des mécanismes de consensus et de minage. Cette évolution marque le début de la blockchain et de son impact dans le domaine financier.[24]

10.2. Blockchain 2.0: « Smart Contracts »

L'épuisement des ressources lié au minage et les limitations en termes de scalabilité de la première génération de blockchain ont incité à élargir le champ d'application de la blockchain au-delà de la simple monnaie. Cette initiative a donné naissance à la deuxième génération de blockchain, symbolisée par Ethereum, qui repose sur de nouveaux concepts tels que les contrats intelligents et des mécanismes de consensus comme la preuve de travail.

La figure 7 illustre le fonctionnement des contrats intelligents. La première étape consiste à formuler le contrat entre deux parties, impliquant que les termes, règles et conditions de l'accord soient acceptés par les deux parties et traduits en code. Aucune modification ne peut être apportée au contrat sans le consentement des parties concernées. Ensuite, le contrat intelligent est déployé sur la blockchain. Dès que les événements spécifiés dans le contrat se produisent, le code s'exécute automatiquement. Par exemple, cela pourrait être l'expiration d'une police d'assurance ou la livraison de marchandises. Une fois l'exécution du code terminée, le contrat transfère automatiquement la valeur au destinataire concerné, permettant un règlement instantané, sûr et efficace. Ce transfert est également enregistré dans la blockchain.[25]

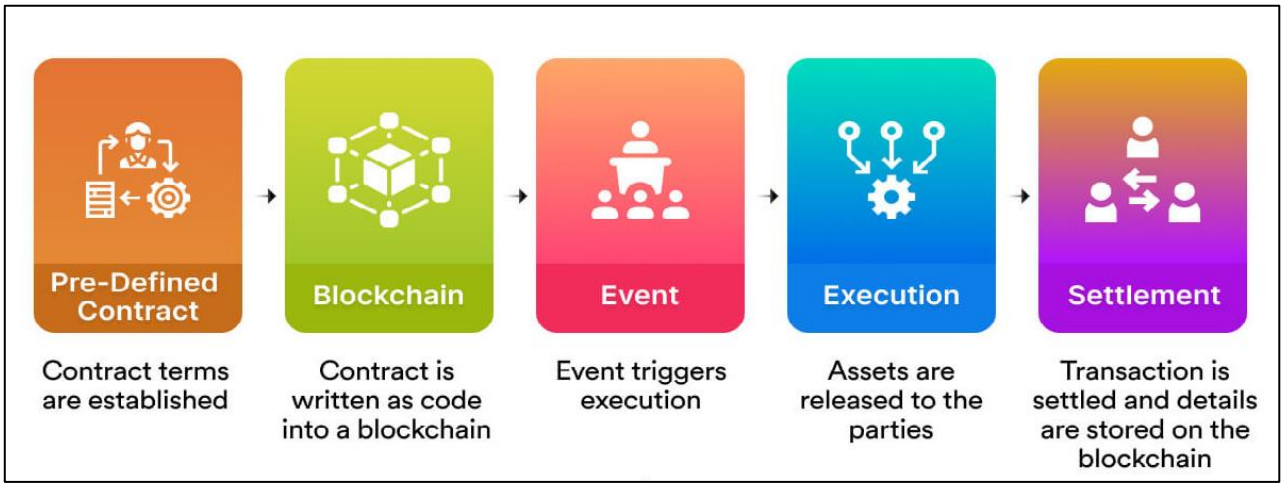

Figure 7– Fonctionnement des contrats intelligents.

10.3. Blockchain 3.0: « Applications Décentralisés »

Blockchain 3.0 représente la génération actuelle de blockchain, visant à résoudre les limites des générations précédentes. Contrairement aux blockchains 1.0 et 2.0, qui présentent des limitations en termes de scalabilité et de confirmation des transactions, la Blockchain 3.0 se concentre sur la viabilité mondiale des cryptomonnaies. En plus des contrats intelligents, cette génération introduit principalement les applications décentralisées (dApps) ,des programmes numériques s'exécutant sur un réseau de blockchain, échappant à toute autorité centrale.[26]

Blockchain 3.0 favorise les transactions inter-chaînes grâce à des techniques telles que le partage, distribuant les données entre les nœuds de la blockchain pour améliorer l'efficacité et la sécurité. Elle utilise les mécanismes de consensus PoW et PoA pour des contrats intelligents plus rapides sans frais de transaction distinct. [27]

Conçue sur le concept "FFM" (Fast, Feeless, Minerless), elle élimine la dépendance à l'égard des mineurs pour vérifier les transactions, garantissant une vitesse élevée de milliers de transactions par seconde.

Bien que Blockchain 3.0 vise à améliorer l'évolutivité, l'interopérabilité, la confidentialité et la durabilité par rapport aux générations précédentes, elle présente des inconvénients tels que la correction de bugs et les mises à jour en raison de sa nature décentralisée. Les mécanismes de consensus appliqués peuvent être comparativement complexes. [24]

10.4. Blockchain 4.0 :

Blockchain 4.0 représente une avancée majeure dans l'évolution de la technologie blockchain. Conçue pour être une plateforme utilisable par les entreprises, elle vise à transformer la blockchain en un courant dominant, permettant la création et l'exécution d'applications. L'innovation clé de la Blockchain 4.0 réside dans sa capacité à intégrer d'autres technologies prospères telles que l'intelligence artificielle, ouvrant ainsi de nouvelles possibilités.

La quatrième génération de blockchain a le potentiel d'atteindre une vitesse de transaction allant jusqu'à un million de transactions par seconde, une réalisation actuellement impossible avec les générations existantes. Cette avancée promet de transformer la manière dont les entreprises utilisent la blockchain, ouvrant la voie à des niveaux de performance sans précédent. Le tableau 1 présente une comparaison entre les différentes générations de blockchain. [24]

Paramètre	Blockchain 1.0	Blockchain 2.0	Blockchain 3.0	Blockchain 4.0
Principe sous-jacent	Technologie du grand livre distribué (DLT)	Contrats intelligents	Application décentralisées (d'Apps)	Blockchain et IA
Mécanisme de consensus	Preuve de travail	Preuve de travail déléguée	Preuve d'enjeu, Preuve d'autorité	Preuve d'intégrité
Vérification	Par les mineurs	Par des contrats intelligents et des mineurs	Mécanisme de vérification intégré via les dApps	Vérification automatisée via Sharding
Evolutivité	Non évolutif	Faiblement évolutif	Evolutif	Hautement évolutif
Interopérabilité	Non interopérable	Non interopérable	Interopérable	Hautement interopérable
intercommunication	Non autorisé	Non autorisé	Autorisé	Autorisé
Vitesse	7 TPS	15 TPS	1000 S TPS	1M TPS
Coût	Coûteux	Moins cher	Plus économique	Coût effectif
Consommation d'énergie	Plus élevé	Modéré	Efficacité énergétique	Très efficace
Exemple	Bitcoin	Ethereum	IOTA, Cardano	SEELE, Unibright
Application	Secteur financier	Secteur non financier	Plateforme d'affaire	Industrie 4.0

Tableau 1– Comparaison entre les différentes générations de blockchain.[28]

11.Domaine d'application de la Blockchain

La blockchain, bien au-delà de son origine avec les cryptomonnaies, offre des applications vastes et diversifiées dans de nombreux domaines. Bien qu'aucune blockchain particulière ne puisse résoudre tous les problèmes, cette technologie demeure une source continue d'innovations, trouvant sa pertinence dans plusieurs secteurs tels que : [29]

- **La finance et les cryptomonnaies :**

La blockchain, pionnière avec le Bitcoin, a rapidement été utilisée comme méthode de paiement et transfert de devises, offrant la possibilité d'effectuer des transactions en quelques secondes.

❖ La santé :

Le secteur de la santé est peut-être l'un des segments où le plus d'inefficacités peuvent être éliminées. L'offre d'une base de données sécurisée avec des règles claires quant aux informations pouvant être partagées – et avec qui – pourrait accélérer considérablement les processus et réduire les coûts.

❖ L'éducation :

Les diplômes et les certificats stockés sur la blockchain pourraient être vérifiables de manière transparente, aidant les employeurs et les institutions à valider plus facilement les qualifications des individus.

❖ Logistique et suivi de la chaîne d'approvisionnement

La logistique pourrait bénéficier grandement de l'aide de la blockchain En effet, l'un des plus grands maux de ce secteur est le manque de communication et de transparence. La blockchain pourrait permettre une identification précise des articles dans la chaîne d'approvisionnement, supprimant tout besoin de transfert d'informations sur papier.

❖ L'immobilier

La blockchain simplifie les transactions immobilières en automatisant les processus de vérification de propriété, de transfert de propriété et d'enregistrement des actes. Cela peut réduire les délais et les coûts associés aux transactions immobilières traditionnelles.

❖ NFT et tokenisation

NFT ou « Non-fungibles Tokens» (jetons non fongibles), ce sont des objets numériques uniques vendus sur la blockchain, impossible à dupliquer. Grâce à la blockchain, les utilisateurs peuvent acheter des contenus numériques uniques (une peinture, un morceau de musique, un objet de collection) et disposer d'une preuve de propriété leur conférant une valeur commerciale.

❖ Utilité gouvernementale et vote électronique

Bien que certains gouvernements craignent que le contrôle ne leur échappe, la blockchain pourrait aider de nombreux Etats à fonctionner de manière plus fluide. La blockchain pourrait être utilisée pour sécuriser les systèmes de vote électronique, éliminant ainsi les préoccupations liées à la fraude électorale et garantissant des élections plus transparentes et justes

❖ Média et droits d'auteur

Aujourd'hui, il est devenu difficile de faire respecter la protection des droits d'auteur sur des supports numériques tels que la musique et les films. La blockchain pourrait être une bonne solution pour les téléchargements de contenu numérique afin de garantir que les artistes et les créateurs de contenu reçoivent leur juste part de compensation. En rendant les fichiers numériques uniques et en les empêchant d'exister à plusieurs endroits en même temps, on pourrait rendre le piratage pratiquement impossible

❖ L'internet des objets :

Avec la sophistication croissante des dispositifs connectés, la blockchain émerge comme une solution prometteuse pour sécuriser les échanges de données au sein de l'Internet des Objets (IoT). La Blockchain assure l'authenticité et la sécurité des informations partagées entre ces objets intelligents en mettant en place des protocoles d'interaction automatisés, assurant ainsi la conservation sécurisée des données, avec un accès restreint exclusivement à l'utilisateur ou aux tiers autorisés.

❖ Energie renouvelable :

La blockchain pourrait révolutionner la distribution d'énergie en permettant aux utilisateurs de vendre leur excédent d'énergie solaire directement à d'autres consommateurs via un réseau décentralisé. Cela pourrait encourager l'utilisation des énergies renouvelables.

❖ La Blockchain pour l'environnement

La blockchain, souvent critiquée pour la consommation énergétique du Bitcoin, se positionne également comme un outil potentiel pour la préservation de l'environnement. Des protocoles innovants comme RubiX cherchent à réduire les émissions de CO2, et plusieurs projets verts fondés sur la blockchain émergent.

Un exemple marquant est Single.Earth, une start-up qui génère des revenus à partir d'arbres sans les abattre. Facilitant la collaboration entre propriétaires fonciers et entreprises, cette plateforme permet aux entreprises de compenser leurs émissions de carbone tout en préservant la biodiversité, en rémunérant les propriétaires fonciers pour la conservation de leurs forêts et marais. Cette approche novatrice souligne le potentiel de la blockchain dans la promotion de la durabilité environnementale.

12.Les plateformes Blockchain

Il existe différents types de plateformes Blockchain qui sont disponibles pour le développement d'applications. Chaque protocole Blockchain adapte les principes de base de la Blockchain pour répondre à des industries ou des applications spécifiques. Quelques exemples de protocoles Blockchain sont fournis dans les sous-sections suivantes : [30]

❖ Hyperledger Fabric

Hyperledger Fabric est un projet open-source avec une suite d'outils et de bibliothèques. Les entreprises peuvent l'utiliser pour créer des applications Blockchain privées rapidement et efficacement. Il s'agit d'un cadre modulaire et polyvalent qui offre des fonctions uniques de gestion d'identité et de contrôle d'accès. Ces caractéristiques le rendent adapté à diverses applications, telles que le suivi et la traçabilité des chaînes d'approvisionnement, les finances commerciales, la fidélisation et les récompenses, ainsi que la compensation et le règlement des actifs financiers.[31]

❖ Ethereum

Ethereum est une plateforme Blockchain décentralisée à code source ouvert que les gens peuvent utiliser pour créer des applications Blockchain publiques. Ethereum Enterprise est conçu pour les cas d'utilisation professionnelle.[25]

❖ Corda

Corda est un projet Blockchain open-source conçu pour les entreprises. Avec Corda, vous pouvez créer des réseaux Blockchain interopérables qui effectuent des transactions en toute confidentialité. Les entreprises peuvent utiliser la technologie des contrats intelligents de Corda pour effectuer des transactions directes, avec de la valeur. La plupart de ses utilisateurs sont des institutions financières.[32].

❖ Quorum

Quorum est un protocole Blockchain open-source dérivé d'Ethereum. Il est spécialement conçu pour être utilisé dans un réseau Blockchain privé, où un seul membre possède tous les nœuds, ou dans un réseau Blockchain de consortium, où plusieurs membres possèdent chacun une partie du réseau.[33]

13.Caractéristiques de Blockchain

13.1. Avantage et Atouts de la Blockchain

La technologie Blockchain repose sur les principales caractéristiques suivantes, comme illustré dans la figure 9 : [13]

❖ La décentralisation :

Un réseau pair à pair où chaque nœud du réseau remplit une ou plusieurs fonctions. L'absence d'autorité centrale qui permet aux utilisateurs d'avoir le total contrôle de leurs données personnelles.

La différence entre un système centralisé et un système décentralisé est illustrée dans la figure 8.

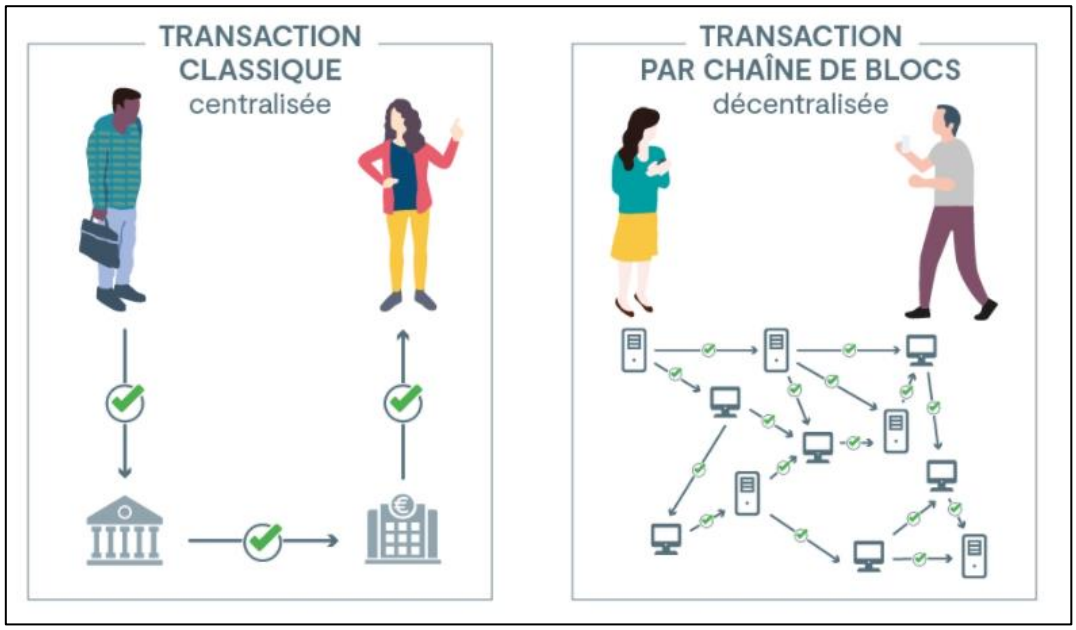

Figure 8- Système Centralisé Vs Système décentralisé. [34]

❖ La transparence :

L'historique des transactions est consultable en permanence par n'importe qui via une connexion internet et un explorateur de Blockchain, le code source d'une Blockchain publique est ouvert et consultable par tous ;

❖ La fiabilité :

La Blockchain repose sur des mécanismes de cryptographie éprouvés et extrêmement robustes, tels que la gestion de binômes clés publiques/clés privées et des fonctions de hachage. De plus, les transactions sont toutes validées par des algorithmes (que l'on appelle consensus) avant d'être partagées au sein de blocs de données ;

❖ L'immuabilité :

La blockchain est un système sécurisé et fiable, une fois insérée dans la Blockchain, une transaction est infalsifiable, y compris par des acteurs malveillants qui participeraient au réseau.

❖ Anonymat :

L'échange de données entre nœuds dans un système blockchain suit un algorithme fixe, les deux parties n'ont donc pas besoin de divulguer leur identité. Au lieu de cela, les règles de procédure dans la blockchain sont utilisées pour se faire confiance.

❖ La Rapidité :

La rapidité des opérations du fait de l'absence de tiers est également un élément compétitif de la blockchain. Il ne faut que quelques secondes pour valider un bloc et donc une transaction, contrairement à un virement bancaire qui prendrait plusieurs heures voire jours.

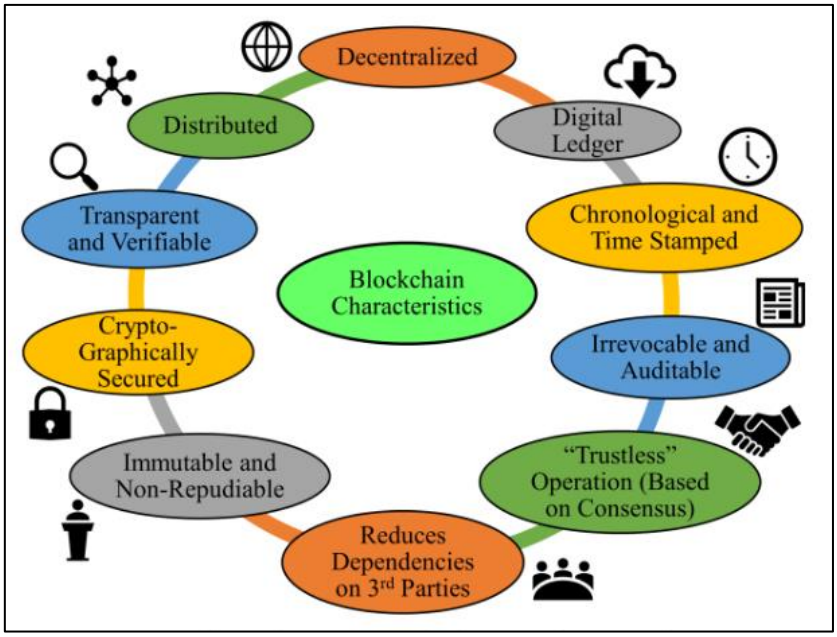

Figure 9- Caractéristiques essentielles de la blockchain. [7]

13.2. Les limites de la blockchain :

L'utilisation de cette technologie révolutionnant le concept de confiance pose tout de même des questions diverses, relatives à la responsabilité sociétale et environnementale, au champ de définition juridique du terme ou encore à la protection des données des utilisateurs.[35]

❖ Limites environnementales :

Les blockchains publiques, comme celle du Bitcoin, consomment énormément d'énergie en raison de leurs opérations distribuées.

❖ Limites liées à la dépendance :

Les utilisateurs dépendent souvent des mineurs, notamment dans des régions où ces activités sont concentrées.

❖ Limites juridiques :

La nature décentralisée de la blockchain pose des défis juridiques, notamment en ce qui concerne la responsabilité et la protection des données des utilisateurs.

❖ Limites sociétales et réglementaires :

L'utilisation de la blockchain nécessite un changement de comportement des utilisateurs et soulève des préoccupations en matière de cybersécurité et d'anonymat.

❖ Limites techniques :

La complexité de la blockchain et le manque de compétences techniques peuvent constituer des obstacles à son adoption.

❖ Limites de capacité et de performance :

La taille croissante de la blockchain, le temps de latence élevé et la consommation d'énergie posent des défis en termes de performance et de scalabilité.

La blockchain présente des avantages significatifs mais doit encore surmonter divers obstacles pour une adoption généralisée.

14.Conclusion

La blockchain, en tant que technologie de stockage et de transmission d'informations, se distingue par sa transparence, sa sécurité et son fonctionnement décentralisé, sans organe central de contrôle. Fondée principalement sur la cryptographie, elle assure la confidentialité des informations numériques au sein du réseau.

Cette innovation a considérablement popularisé les nouvelles monnaies virtuelles, marquant une avancée pionnière dans le domaine des applications transactionnelles. En adoptant un réseau pair à pair, la blockchain, avec son mécanisme de consensus collectif et l'utilisation d'un grand livre de compte public décentralisé, établit la confiance, la responsabilité et la transparence, tout en rationalisant les processus d'affaires.

Face aux défis persistants en matière de confidentialité et de sécurité, des méthodes innovantes ont été recherchées pour renforcer ces aspects. C'est dans ce contexte que le Zero-Knowledge Proof (ZKP) intervient. Cette méthode, visant à prouver des affirmations sans divulguer d'informations supplémentaires, représente une avancée majeure pour assurer la sécurité des transactions sur la blockchain. En intégrant le ZKP dans les applications blockchain, on peut améliorer la protection des données et ouvrir de nouvelles voies pour l'innovation et l'adoption de cette technologie dans divers domaines. Nous explorerons en détail le fonctionnement et les applications du ZKP dans le chapitre suivant.

CHAPITRE II

L’AUTHENTIFICATION

CHAPITRE II : L'AUTHENTIFICATION

1. Introduction

Dans l'ère numérique contemporaine, la sécurité informatique est devenue une préoccupation primordiale pour les organisations, les entreprises et les institutions. L'évolution rapide des technologies numériques a considérablement augmenté les risques liés à la sécurité des données et des systèmes, soulignant ainsi l'urgence de mettre en place des mesures robustes pour protéger les informations sensibles et garantir le bon fonctionnement des environnements informatiques.

Au centre de ces mesures de sécurité se trouve l'authentification, qui représente la première ligne de défense contre les menaces cybernétiques. L'authentification est le processus crucial permettant de vérifier si un utilisateur est bien celui qu'il prétend être. Elle joue un rôle essentiel en agissant comme une barrière initiale contre les intrusions et les accès non autorisés. Son objectif principal est de confirmer l'identité des utilisateurs cherchant à accéder aux systèmes, aux réseaux et aux données sensibles, assurant ainsi que seules les personnes légitimes et autorisées puissent franchir le périmètre de sécurité d'une organisation.

Il est toutefois essentiel de faire la distinction entre l'authentification et l'autorisation. Alors que l'authentification se concentre sur la confirmation de l'identité numérique de l'utilisateur, l'autorisation intervient après cette étape pour définir les droits et les permissions attribués à cet utilisateur une fois authentifié. Ces droits incluent l'accès à des ressources spécifiques, les privilèges pour effectuer certaines actions sur ces ressources, ainsi que la possibilité d'utiliser des fonctionnalités spécifiques des systèmes informatiques.

La structure de l'authentification comprend plusieurs éléments clés : l'utilisateur présente des informations d'identification, le système d'authentification vérifie ces informations en les comparant avec celles stockées dans une base de données sécurisée, et enfin, le système décide si l'accès doit être accordé ou refusé. Cette structure garantit que seules les identités vérifiées peuvent accéder aux ressources protégées.

Dans ce chapitre, nous examinerons de manière approfondie l'importance, les méthodes et les technologies de l'authentification dans le contexte complexe de la sécurité informatique. Nous aborderons les sujets suivants : les facteurs d'authentification, les types d'authentification, les méthodes d'authentification, et les protocoles d'authentification. Nous fournirons également des recommandations et des solutions pour renforcer la protection des systèmes et des données contre les menaces cybernétiques.

2. Définition

L'authentification est un processus essentiel au sein des systèmes informatiques, permettant de vérifier l'identité d'une personne ou d'un appareil. Cela se fait en comparant les informations fournies par l'utilisateur avec celles stockées dans une base de données.

Seul l'administrateur du système peut définir les droits d'accès et configurer les paramètres en conséquence. Un utilisateur, à travers son identifiant et son mot de passe, n'a accès qu'aux ressources autorisées.[36]

L'objectif initial est de prévenir l'accès non autorisé à des données sensibles, que ce soit dans des bases de données, des réseaux ou d'autres ressources confidentielles. Les différents types d'authentification reposent sur des facteurs qui servent à confirmer l'identité de l'utilisateur. Cela inclut des éléments tels que les mots de passe, les clés d'authentification à deux facteurs, les certificats numériques ou encore des données biométriques comme les empreintes digitales ou la reconnaissance faciale. [37]

En résumé, l'authentification complète le processus d'identification en assurant que l'utilisateur, l'application ou l'équipement est bien celui qu'il prétend être. La méthode d'authentification choisie dépend des exigences de sécurité et de la nature des interactions et des ressources du système informatique.

3. Facteurs d'authentification

Il existe quatre (4) facteurs d'authentifications qui peuvent être utilisés dans le processus d'autorisation d'accès à des ressources bloquées et sécurisées :

❖ Ce que l'on connait (facteur mémoriel) :

Une information que l'utilisateur a mémorisée et que lui seul connait (exemple : un mot de passe, un nom).

❖ Ce que l'on possède (facteur matériel) :

Une information que seule l'utilisateur possède et enregistre dans un support (exemple : une clé USB).

❖ Ce que l'on est (facteur corporel) :

Une information qui caractérise l'utilisateur avec une empreinte qui lui est propre (exemple : voix, pupille, empreinte digitale)

❖ Ce que l'on sait faire (facteur réactionnel) :

Une information ou un geste que seul l'utilisateur peut produire (exemple : une signature)

Ces différents facteurs d'authentification utilisés seuls ou en combinaison, permettent d'identifier l'utilisateur et d'apporter la preuve que cette personne est bien autorisée à accéder à un compte sécurisé. [38]

4. Les méthodes d'authentifications

❖ Authentification par mot de passe

Ce type d'authentification exige que le fournisseur se rappelle ce qu'il sait. Il y a deux parties dans cette méthode. Premièrement, le fournisseur entre le nom d'utilisateur et, deuxièmement, le mot de passe. Le mot de passe est la combinaison secrète de mots et de chiffres que le fournisseur connaît.[39]

❖ **Authentification par carte à puce**

L'authentification par carte à puce est un facteur qu'un utilisateur a. Une carte à puce est une carte de la taille d'une carte de crédit qui a un certificat intégré utilisé pour identifier le titulaire. L'utilisateur peut insérer la carte dans un lecteur de carte à puce pour authentifier l'individu. Les cartes à puce sont couramment utilisées avec un NIP fournissant l'authentification multifactorielle. En d'autres termes, l'utilisateur doit avoir quelque chose (la carte à puce) et savoir quelque chose (le PIN). [40]

❖ **Authentification biométrique**

L'authentification biométrique est une méthode qui identifie un utilisateur et/ou vérifie son identité en fonction de la mesure de ses caractéristiques physiologiques ou comportementales uniques. La biométrie physiologique est l'empreinte digitale, la reconnaissance faciale, la géométrie de la main. La biométrie comportementale est la reconnaissance vocale, la démarche et le balayage de la signature. Les empreintes digitales et les empreintes de mains sont la méthode biométrique la plus utilisée aujourd'hui. [41]

❖ **Authentification par certificat numérique**

Un certificat numérique est une technologie de chiffrement qui fonctionne comme la version Internet d'un passeport. À l'aide d'une clé publique et d'une clé privée, les certificats numériques permettent essentiellement de s'assurer que le message provient d'une personne en particulier. Le certificat numérique authentifie l'identité de l'expéditeur pour assurer une communication plus sûre et prévenir la fraude sur Internet. Les plus grands avantages de l'authentification numérique basée sur les certificats sont la confidentialité. En cryptant vos communications, courriels, ouvertures de session ou transactions bancaires en ligne, les certificats numériques protègent les données privées et empêchent les renseignements d'être vus par des yeux involontaires. Les systèmes de certificats numériques sont également conviviaux, fonctionnent généralement automatiquement et nécessitent un minimum d'action ou de participation de la part des expéditeurs ou des destinataires. [42]

5. Les types d'authentification

Les différentes méthodes d'authentification comprennent l'authentification simple, l'authentification unique, l'authentification forte et l'authentification biométrique.

5.1. Authentification Simple

L'authentification simple repose sur l'utilisation d'un seul facteur pour vérifier l'identité d'un utilisateur. Ce facteur peut être quelque chose que l'utilisateur sait, comme un mot de passe, un code PIN, ou une question de sécurité. Cette méthode est la plus basique et la plus couramment utilisée pour sécuriser l'accès aux systèmes informatiques. Cependant, elle présente des risques car si ce facteur unique est compromis, cela peut entraîner un accès non autorisé aux données et aux ressources sensibles. [43]

5.2. Authentification Unique

L'authentification unique (Single Sign-On - SSO) permet à un utilisateur de s'authentifier une seule fois pour accéder à plusieurs applications ou services informatiques sans avoir à saisir ses identifiants à chaque fois. Une fois l'utilisateur authentifié, un jeton d'authentification est généré et utilisé pour valider son accès à différentes ressources. Cela simplifie l'expérience de l'utilisateur en réduisant le nombre de fois où il doit saisir ses identifiants, tout en renforçant la sécurité en évitant la multiplication des mots de passe. Par exemple, en utilisant Google SSO, un utilisateur peut se connecter une fois à son compte Google et accéder ensuite à des services comme YouTube, Gmail, Google Drive, ainsi qu'à des applications d'entreprise telles que Slack ou Salesforce, sans avoir à entrer ses identifiants pour chaque application.[44]

5.3. Authentification Forte

L'authentification forte est une méthode plus avancée et sécurisée qui combine au moins deux facteurs différents pour vérifier l'identité de l'utilisateur [10]. En utilisant deux facteurs ou plus, l'authentification forte réduit considérablement les risques de compromission des comptes et renforce la sécurité globale des systèmes informatiques :

5.3.1. Authentification à deux facteurs (2FA) :

L'authentification à deux facteurs ajoute une couche supplémentaire de sécurité en exigeant un second facteur de vérification. Cela rend plus difficile pour les attaquants de compromettre un compte, car ils doivent contourner deux méthodes d'authentification distinctes. Les facteurs de vérification secondaires peuvent être un mot de passe à usage unique provenant d'une application d'authentification, un numéro de téléphone pour recevoir un code SMS, ou même des éléments biométriques tels que des empreintes digitales ou la reconnaissance faciale. Bien que cette méthode offre une sécurité accrue, elle peut être perçue comme moins pratique pour les utilisateurs en raison de la nécessité de saisir un deuxième facteur d'authentification.[45]

5.3.2. Authentification multifactorielle (MFA) :

L'authentification multifacteur (MFA) est une méthode hautement fiable qui utilise plusieurs facteurs de vérification pour valider l'identité de l'utilisateur. Contrairement à l'authentification à deux facteurs, qui utilise toujours deux facteurs, l'authentification MFA peut inclure deux ou trois facteurs, voire plus, selon les besoins de sécurité. Les facteurs supplémentaires peuvent inclure des données biométriques, la confirmation axée sur l'appareil, des mots de passe supplémentaires, ou des informations basées sur le comportement de l'utilisateur. Cette approche renforce considérablement la sécurité en ajoutant des couches d'authentification supplémentaires, ce qui rend la compromission des comptes plus difficile pour les attaquants.[46]

En résumé, l'authentification simple est basée sur un seul facteur, l'authentification unique permet un accès multiple avec une seule authentification, l'authentification biométrique s'appuie sur la vérification d'une caractéristique physique de l'utilisateur, tandis que l'authentification forte combine plusieurs facteurs pour garantir une sécurité accrue.

6. Les Protocoles d'authentification

Les Protocoles d'Authentification proposent diverses méthodes pour réaliser l'authentification, bien que leur mise en œuvre puisse varier en termes de robustesse et de processus impliqués. [44].

Les protocoles ou les mécanismes d'authentification décrits dans cette partie, ont tout d'abord été des protocoles de la deuxième couche du modèle OSI (couche liaison), puisqu'ils ont été initialisés par le Protocole PPP qui permet l'ouverture de session sur le RTC (Le réseau téléphonique commuté). [47]

Actuellement, ils sont également utilisés dans la couche réseau grâce au passage de PPP à PPPoA (over ATM) et PPPoE (over Ethernet) qui sont principalement utilisés pour ouvrir des connexions ADSL. Cependant, ces mécanismes sont les briques de nombreux serveurs et applications d'authentifications comme RADIUS, TACACS+, Kerberos…etc. [48]

6.1.1. Le protocole d'authentification par mot de passe (PAP)

Le protocole PAP (Password Authentication Protocol) est un protocole d'authentification, utilisé avec le Protocole PPP (Point to Point Protocol), permet d'identifier un utilisateur auprès d'un serveur PPP en vue d'une ouverture de connexion sur le réseau [49]. Après une phase de synchronisation entre le client et le serveur PPP, comme illustré dans la figure 10 , le processus d'authentification se fait en deux étapes :

- ✓ Le client envoie son nom PAP ainsi que son mot de passe en clair.
- ✓ Le serveur qui détient une table de noms d'utilisateurs et de mots de passe vérifie que le mot de passe correspond bien à l'utilisateur et valide ou rejette la connexion.[50]

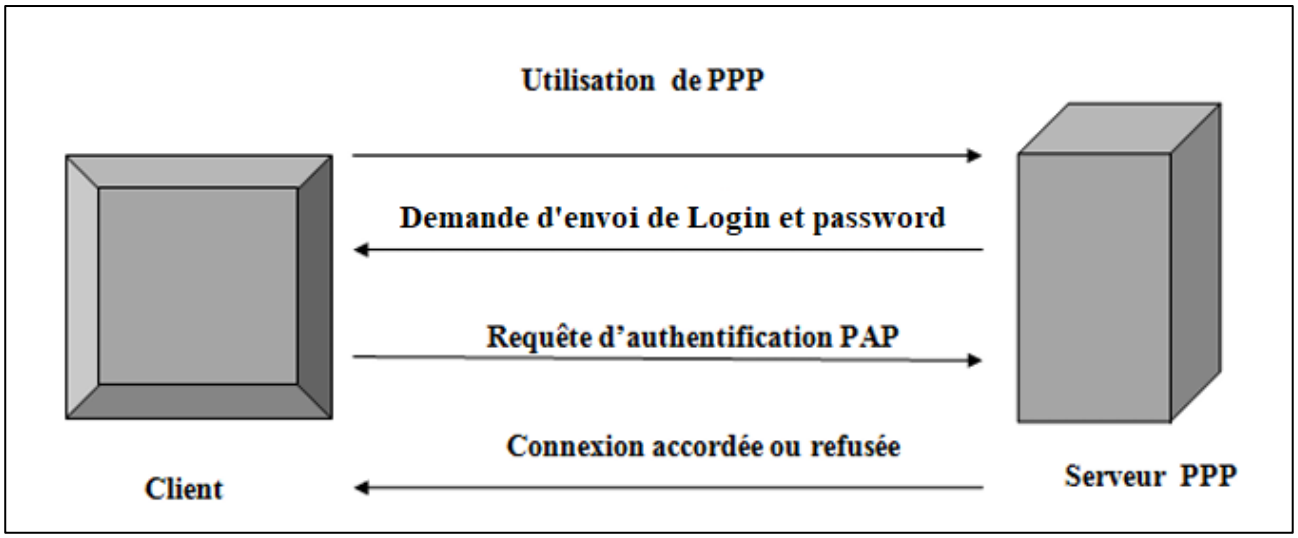

Figure 10 - Phase de synchronisation de protocole PPP. [51]

6.1.2. Le protocole d'authentification par défi-réponse (CHAP)

Contrairement au Protocole PAP, le Protocole CHAP (Challenge Handshake Authentification Protocole) permet une authentification sécurisée par hachage MD5 (Message Digest 5).

MD5 et une fonction de hachage cryptographique permettant d'obtenir l'empreinte numérique d'un message à partir du quelle il est impossible de retrouver le message original. Ainsi, en envoyant

l'empreinte du mot de passe au serveur, le client peut montrer qu'il connaît bien le mot de passe sans avoir à réellement l'envoyer sur le réseau. Après le même type de synchronisation que pour le Protocole PAP, le mécanisme d'authentification est basé sur un CHALLENGE en 3 étapes, comme illustré dans la figure 11 :

- ✓ Le serveur envoie au client un nombre aléatoire de 16 bits ainsi qu'un compteur incrémenté à chaque envoi.
- ✓ Le client génère une empreinte MD5 de l'ensemble constitué reçu puis il envoie cette empreinte.
- ✓ Le serveur calcule également de son côté l'empreinte MD5 grâce au mot de passe du client stocké localement puis il compare son résultat à l'empreinte envoyée par le client. Si les deux empreintes sont identiques, le client est bien identifié et la connexion peut s'effectuer sinon, elle est rejetée. [52]

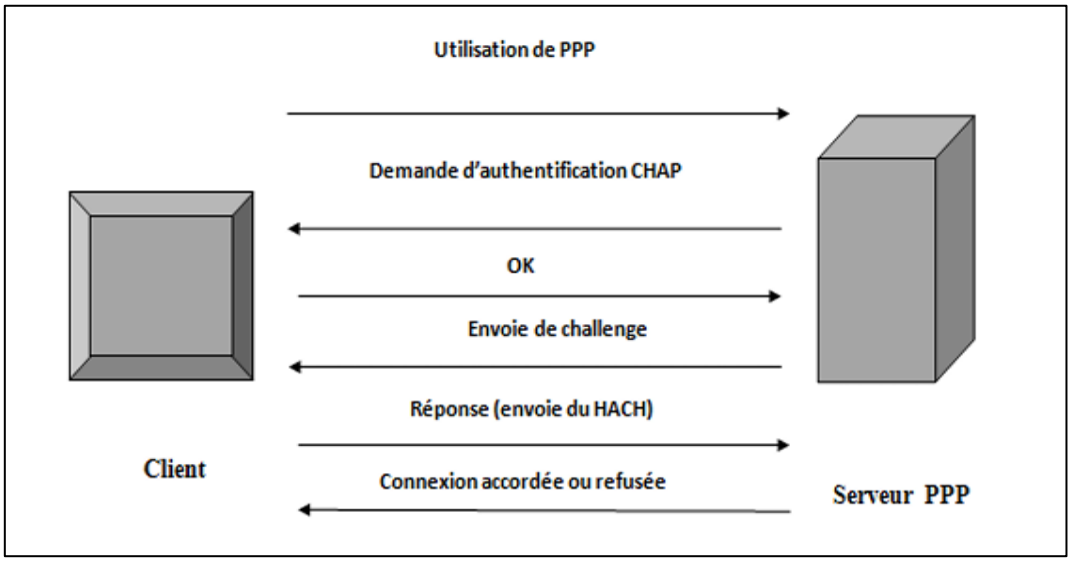

Figure 11– les étapes d'authentification de protocole CHAP. [51]

Ce mécanisme d'authentification présente deux avantages pour CHAP : tout d'abord, si la requête d'authentification envoyée par le client est interceptée, elle ne pourra pas être rejouée, car chaque empreinte calculée par le client est unique et envoyée par le serveur. Ensuite, pendant une session établie par le protocole CHAP, le serveur envoie régulièrement des défis au client pour confirmer son identité. Cette mesure renforce la sécurité en empêchant les tentatives de détournement de session, garantissant ainsi l'intégrité des échanges entre le client et le serveur.

6.1.3. Le protocole MS-CHAP

MS-CHAP ou (Microsoft Challenge Handshake Authentification Protocol) est la version Microsoft de protocole CHAP. Plus qu'une simple version prioritaire, MS-CHAP apporte des améliorations à CHAP.

Un des principaux inconvénients de CHAP est que le serveur doit détenir les mots de passe des utilisateurs en clair pour pouvoir vérifier l'empreinte MD5 envoyée par les clients, ce qui constitue une vulnérabilité potentielle en cas de compromission du serveur. Pour remédier à cette faiblesse, le Protocole MS-CHAP intègre une fonction de hachage propriétaire permettant de stocker sur le serveur un hash intermédiaire du mot de passe. [53]

Ainsi, en travaillant uniquement avec cet hash intermédiaire au lieu du mot de passe, le client et le serveur peuvent réaliser le même type de procédure que celle du CHAP, ainsi, le mot de passe en clair n'a plus besoin d'être stocké sur le serveur. [54]

Malgré l'avancée du Protocole MS-CHAP par rapport à CHAP, Microsoft créa une seconde version su Protocole (MS-CHAP-v2) pour résoudre deux principales faiblesses de MS-CHAP-v1, d'une part le fait que le client ne puisse pas vérifier l'authenticité du serveur sur lequel il veut se connecter et d'autre part que l'algorithme de hachage propriétaire utilisé soit très vulnérable à des attaques par brute-force. [55]

Voici le fonctionnement du processus d'authentification mutuelle fournit par MS-CHAP-v2 :

- ✓ Le serveur d'accès disant envoie une demande de vérification au client contenant une identification de session I et une chaine C1 générée aléatoirement.
- ✓ Le client envoie alors une réponse contenante : son nom d'utilisateur, une chaîne aléatoire C2 et un hash de l'ensemble formé par la chaîne C1, l'identificateur de session I et son mot de passe.
- ✓ Le serveur vérifie la réponse du client et il renvoie une réponse contenante : une chaîne indiquant le succès ou l'échec de l'authentification, et un hash de l'ensemble formé par 3 éléments : la chaîne C2, l'identificateur de session I et son mot de passe.
- ✓ Le client vérifie à son tour la réponse d'authentification et établit la connexion en cas de réussite.

6.1.4. Le standard 802.1X /Protocole d'authentification extensible (EAP)

Ce standard 802.1X /EAP Ce standard a été mis au point par l'IEEE en juin 2001, il a comme objectif de réaliser une authentification de l'accès au réseau au moment de la connexion physique à ce dernier et ce en s'appuyant sur le protocole EAP (Extensible Authentication Protocol) il ne nécessite que très peu de ressources pour fonctionner, dans le cas d'un réseau sans fil, c'est le point d'accès qui joue le rôle de contrôleur d'accès. [56]

Cette authentification intervient avant tout mécanisme d'auto configuration (ex. DHCP, PXE...).

Dans la plupart des cas, le service autorisé en cas de succès est le service Ethernet. L'objectif de ce standard est donc uniquement de valider un droit d'accès physique au réseau, indépendamment du support de transmission utilisé, et en s'appuyant sur des mécanismes d'authentification existants. [57]

Dans le fonctionnement du protocole, les trois entités qui interagissent, illustrés dans la figure 12, sont : le système à authentifier le système authentificateur et un serveur d'authentification. Le système authentificateur contrôle une ressource disponible via le point d'accès physique au réseau, nommé PAE (Port Access Entity). [58]

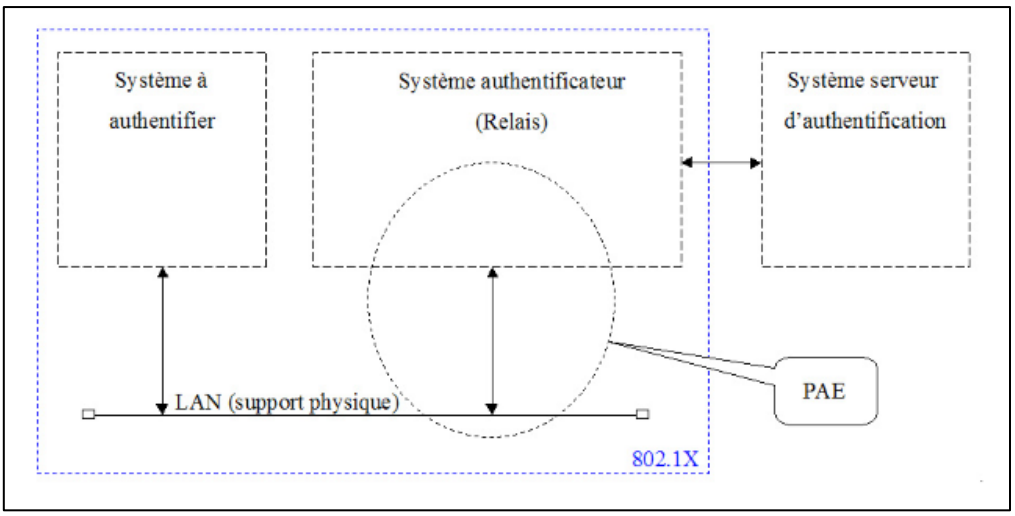

Figure 12– Les trois entités qui interagissent dans 802.1X. [59]

6.1.5. Le protocole KERBEROS :

Kerberos est un protocole d'authentification réseau développé par le Massachusetts Institute of Technology et standardisé par l'IETF, il porte le nom grec du Cerbère gardien des Enfers le chien à trois têtes. Ce protocole repose sur un mécanisme de clés secrètes (chiffrement symétrique) et l'utilisation de tickets, et non de mots de passe en clair, évitant ainsi le risque d'interception frauduleuse des mots de passe des utilisateurs. [60]

L'objectif fondamental de Kerberos est de fournir une double fonctionnalité : sécuriser les échanges au sein d'un réseau non sécurisé et garantir une authentification fiable des utilisateurs.

Pour atteindre cet objectif, Kerberos repose sur deux entités principales :

- ✓ *Serveur d'Authentification (AS - Authentication Server)* : Cette entité gère l'aspect authentification du processus pour les clients. Seul le serveur d'authentification peut permettre au client de communiquer avec le Serveur de Distribution de Tickets (TGS) en lui délivrant un ticket d'accès.
- ✓ *Serveur de Distribution de Tickets (TGS - Ticket Granting Server)* : Le TGS gère les demandes d'accès aux services des clients déjà authentifiés. Une fois que le client est authentifié AS, il peut utiliser le ticket d'accès délivré pour accéder aux services du TGS.

Ces deux entités, le AS et le TGS, sont regroupées au sein d'une infrastructure appelée le Centre de Distribution de Clés (KDC - Key Distribution Center). Typiquement, le KDC est hébergé sur un même serveur, ce qui simplifie la gestion et la sécurité du processus d'authentification dans le système Kerberos. [61]

6.1.6. Les protocoles triple-A :

Triple-A ou AAA est une abréviation de l'expression en englais Authentication (authentification), Authorization (autorisation) et Accounting (journalisation ou comptabilisation) : c'est un modèle de protocole de sécurité .il englobe trois fonctions principales de sécurité : l'authentification, l'autorisation et la comptabilisation. L'authentification vérifie qu'une entité est bien celle qu'elle prétend être. L'autorisation détermine les actions que l'utilisateur authentifié peut effectuer, ainsi que les ressources et services auxquels il a accès. La comptabilisation collecte et enregistre les

informations sur les actions de l'utilisateur depuis son authentification jusqu'à la fin de sa session, en mesurant également les ressources consommées. [62]

Les deux principaux protocoles pour la communication entre un client et un serveur triple-A sont RADIUS et TACACS+. Toutefois nous pouvons mentionner d'autres, notamment DIAMETER et TACACS.

6.1.7. Le protocole RADIUS :

RADIUS (Remote Authentification Dial In User Service) est un protocole d'authentification client/serveur habituellement utilisé pour l'accès distant, défini par la RFC 2865. Ce protocole permet de sécuriser les réseaux contre des accès à distance non autorisés. Ce protocole est indépendant du type de support utilisé. [63]

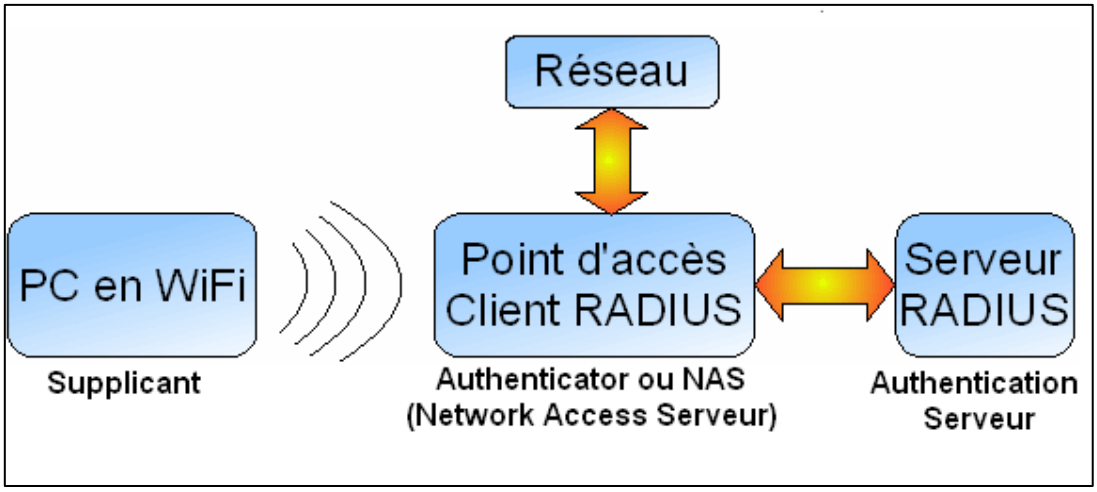

Figure 13- Le protocole RADIUS [64]

Le protocole Radius repose principalement sur un serveur (serveur Radius), relié à une base d'identification (fichier local, base de données, annuaire LDAP, etc.) et un client Radius, appelé NAS (Network Access Server), faisant office d'intermédiaire entre l'utilisateur final et le serveur, comme illustré dans la figure 13. Le mot de passe servant à authentifier les transactions entre le client Radius et le serveur est chiffré et authentifié grâce à un secret partagé. Il est à noter que le serveur Radius peut faire office de proxy, c'est-à-dire transmettre les requêtes du client a d'autres serveurs Radius.[65]

Le fonctionnement de Radius se déroule généralement ainsi : un utilisateur envoie une requête au NAS pour autoriser une connexion à distance ; le NAS transmet la demande au serveur Radius ; le serveur Radius consulte la base de données d'identification pour déterminer le type de scénario d'authentification requis. Si le scénario actuel convient, l'utilisateur est authentifié, sinon une autre méthode d'authentification est demandée à l'utilisateur. [66]

6.1.8. Le protocole TACACS :

TACACS (Terminal Access Controller Access-Control System) est un protocole d'authentification distant utilisé pour communiquer avec un serveur d'authentification, généralement utilisé dans des réseaux UNIX. TACACS permet à un serveur d'accès distant de communiquer avec

un serveur d'authentification dont l'objectif est de déterminer si l'utilisateur a le droit d'accéder au réseau. Sa définition complète est faite dans la RFC 1492. [67]

6.1.9. Le protocole TACACS+ :

TACACS+ (Terminal Access Controller Access-Control SystemPlus) est la dernière version du protocole TACACS. Développé à l'origine par BBN puis repris par Cisco, il a été étendu une première fois avec XTACACS (eXtended TACACS). TACACS+ utilise le protocole TCP et le port 49 pour son transport, contrairement à TACACS qui s'appuie sur UDP. Il gère séparément les trois fonctions AAA (Authentication, Authorization, Accounting). [68]

6.1.10. Le protocole Schnorr :

Le protocole Schnorr, conçu par Claus-Peter Schnorr en 1989, est un schéma cryptographique basé sur la difficulté du problème logarithmique discret dans des groupes cycliques. Il est largement reconnu pour sa sécurité et son efficacité. [69]

La simplicité et l'efficacité computationnelle du protocole Schnorr, ainsi que sa robustesse contre les attaques cryptographiques, en font un choix attractif pour notre application.

7. Les Protocoles d'authentification Intégrables à la Blockchain

Dans le domaine de la blockchain, l'authentification joue un rôle crucial pour sécuriser les transactions et garantir l'identité des utilisateurs. Plusieurs protocoles d'authentification sont utilisés, chacun offrant des avantages spécifiques. Voici une vue d'ensemble détaillée de huit de ces protocoles :

7.1.1. PKI (Infrastructure à Clés Publiques)

La PKI, basée sur l'utilisation de paires de clés publiques et privées, est fondamentale pour sécuriser les transactions et les identités sur la blockchain. Associant chaque utilisateur à une clé publique, elle garantit que seules les personnes autorisées peuvent accéder et effectuer des transactions sur le réseau décentralisé.[70]

7.1.2. OAuth (Open Authorization)

OAuth est un protocole d'authentification qui permet à un utilisateur de partager de manière sécurisée ses ressources avec des applications tierces sans divulguer ses identifiants. En échangeant des jetons d'accès entre l'application cliente et le fournisseur de services, OAuth facilite l'accès contrôlé aux données des utilisateurs. [71]

7.1.3. OpenID Connect

OpenID Connect, construit sur OAuth 2.0, offre une couche d'identité standardisée pour les interactions authentifiées. Il permet aux utilisateurs de se connecter à divers sites et applications avec un seul compte, tandis que les fournisseurs d'identité gèrent l'authentification de manière sécurisée. [71]

7.1.4. SAML (Security Assertion Markup Language)

SAML facilite l'échange sécurisé d'informations d'authentification entre les parties, souvent utilisé dans les fédérations d'identité. Il fournit un cadre standardisé pour l'authentification unique (SSO) et la gestion des autorisations dans les environnements distribués.[72]

7.1.5. JWT (JSON Web Tokens)

JWT est un format de jeton compact largement utilisé pour l'authentification entre deux parties. Il est particulièrement adapté aux systèmes basés sur des API RESTful, offrant un moyen sécurisé de transférer des informations d'identité entre les composants d'une application distribuée.[73]

7.1.6. DID (Decentralized Identifiers)

Les DID offrent des identifiants auto-souverains basés sur la cryptographie, permettant aux individus et aux entités de contrôler leur identité en ligne de manière décentralisée. Ils garantissent l'authenticité et la portabilité des identités numériques dans les écosystèmes blockchain.[74]

7.1.7. SSI (Self-Sovereign Identity)

Le concept SSI met l'accent sur l'individu en tant que détenteur souverain de son identité, tirant parti des DID et d'autres technologies pour assurer la gestion sécurisée et privée des informations d'identification. Il offre un cadre pour une identité numérique autonome et résiliente.[75]

7.1.8. ZKP (Zero-Knowledge Proofs)

Les ZKP permettent de prouver qu'une déclaration est vraie sans révéler d'informations supplémentaires, offrant ainsi un mécanisme d'authentification robuste tout en préservant la confidentialité des données. Ils sont utilisés pour établir des preuves d'authentification sans divulguer d'informations sensibles.[76]

Ces protocoles d'authentification offrent un large éventail de fonctionnalités et sont adaptés à divers scénarios d'application dans les systèmes basés sur la blockchain.

Le ZKP sera le protocole choisi pour une étude plus approfondie dans le prochain chapitre.

8. Conclusion

En conclusion, l'authentification occupe une place centrale dans la sécurisation des systèmes informatiques et des données sensibles. À travers l'étude des différentes familles d'authentification, nous avons pu comprendre l'importance de choisir des méthodes adaptées aux besoins de sécurité tout en tenant compte de la convivialité pour les utilisateurs. L'authentification à facteur unique offre une simplicité d'utilisation mais présente des risques de sécurité plus élevés, tandis que l'authentification multifactorielle ou l'authentification forte renforce la sécurité en introduisant des couches supplémentaires de vérification.

Les protocoles d'authentification jouent également un rôle crucial, avec des solutions telles que Kerberos qui visent à sécuriser les échanges sur les réseaux non sécurisés tout en garantissant une authentification fiable des utilisateurs. En intégrant des pratiques d'authentification robustes, des protocoles sécurisés et une sensibilisation continue à la sécurité informatique, les organisations peuvent renforcer leur posture de sécurité et réduire les risques liés aux accès non autorisés et aux intrusions malveillantes.

Il est essentiel pour les professionnels de la sécurité informatique de rester informés des dernières avancées et des meilleures pratiques en matière d'authentification afin de garantir une protection optimale des systèmes et des données contre les cybermenaces en constante évolution.

En lien direct avec les défis de l'authentification, les Zero Knowledge Proofs (ZKP) offrent une approche innovante pour renforcer la sécurité. En poursuivant notre exploration des technologies de sécurité avancées, nous allons maintenant aborder le concept de ZKP.

Cette méthode cryptographique révolutionnaire permet à une partie de prouver à une autre partie qu'elle connaît une valeur spécifique sans révéler aucune information supplémentaire. Les ZKP promettent de renforcer considérablement la confidentialité et la sécurité des transactions numériques, ouvrant de nouvelles perspectives pour la protection des données sensibles et l'authentification sécurisée.

CHAPITRE III :

ZERO-KNOWLEDGE PROOF (ZKP)

CHAPITRE III : ZERO-KNOWLEDGE PROOF

1. Introduction

Dans le domaine de la sécurité informatique, la découverte et la divulgation des vulnérabilités sont cruciales pour la protection des systèmes et des utilisateurs. Une vulnérabilité informatique représente une faille susceptible d'être exploitée par des individus malveillants pour des actions non autorisées, mettant ainsi en danger la confidentialité, l'intégrité et la disponibilité des données. Face à ce défi, la question de la divulgation responsable des vulnérabilités est centrale au sein de la communauté de la sécurité informatique.

Cette problématique suscite un débat entre deux approches opposées : la divulgation complète et la divulgation responsable. La première préconise une communication immédiate et publique de toute vulnérabilité découverte, tandis que la seconde favorise une communication restreinte initialement à l'entreprise ou aux utilisateurs concernés, afin de minimiser les risques pour ces derniers. Chacune de ces méthodes présente des avantages et des inconvénients, en fonction des parties prenantes impliquées, des implications éthiques et des conséquences sur la sécurité des systèmes.

Jusqu'à récemment, aucune solution n'était en mesure de prouver l'existence d'une vulnérabilité sans la divulguer intégralement, laissant ainsi les chercheurs en sécurité dans une position délicate lorsqu'ils découvraient de telles failles. Cependant, l'émergence des technologies de preuve à divulgation nulle de connaissance (Zero-knowledge proof - ZKP) ouvre de nouvelles perspectives dans ce domaine. Ces technologies offrent la possibilité de prouver la validité d'une déclaration sans révéler d'autres informations que cette validité, ouvrant ainsi la voie à une approche plus sécurisée de la divulgation des vulnérabilités.

Dans ce chapitre, nous nous concentrerons sur les fondements de la preuve à divulgation nulle de connaissance (ZKP). Nous expliquerons son fonctionnement et ses caractéristiques distinctives, ainsi nous fournirons des exemples concrets pour illustrer ses applications. L'objectif est de mettre en évidence le rôle crucial de ZKP dans la sécurité informatique, en illustrant comment elle peut être utilisée pour protéger les données sensibles et sécuriser les transactions.

2. Les origines du Zero-Knowledge Proof (ZKP)

L'idée de la preuve à connaissance nulle a été introduite pour la première fois dans un article fondateur publié en 1985 par les cryptographes Shafi Goldwasser, Silvio Micali et Charles Rackoff. Leur travail, intitulé "*The Knowledge Complexity of Interactive Proof-Systems*", a fourni les bases théoriques de ZKP en définissant le concept de "complexité de la connaissance". Ils ont démontré comment il est possible de prouver qu'une déclaration est vraie sans révéler pourquoi elle est vraie, établissant ainsi les principes fondamentaux des preuves à connaissance nulle [77].

Après la publication initiale, d'autres chercheurs ont rapidement poursuivi les travaux dans ce domaine. En 1986, Manuel Blum, Paul Feldman et Silvio Micali ont proposé le premier exemple concret de preuve à connaissance nulle pour le problème de la couleur de graphes [78].

En 1987, Oded Goldreich, Silvio Micali et Avi Wigderson ont généralisé les concepts en introduisant des preuves à connaissance nulle pour tous les langages dans NP (classe de complexité non déterministe polynomiale), démontrant ainsi l'universalité de ces protocoles.[79]

Les avancées théoriques dans le domaine des preuves à connaissance nulle ont rapidement trouvé des applications pratiques, notamment dans les systèmes d'authentification. En utilisant les fondements établis par les chercheurs, l'intégration de ces preuves dans des systèmes concrets a marqué une étape clé dans leur développement. Cette transition a permis de relier la théorie à des solutions pratiques, ouvrant ainsi la voie à des innovations futures dans divers domaines technologiques.

3. Définition de ZKP

ZKP ou Zero-Knowledge Proof est un concept cryptographique qui permet à un prouveur de démontrer la validité d'une affirmation à un vérificateur sans révéler aucune information supplémentaire autre que la véracité de cette affirmation. La figure 14 présente un exemple du concept de ZKP .[80]

En d'autres termes, une partie assure à une autre qu'une déclaration est vraie sans montrer autre chose que la véracité de la déclaration. Le prouveur et le vérificateur utilisent une valeur numérique, qui agit comme nombre secret pour le prouveur. Fondamentalement, le prouveur P propose des problèmes mathématiques intensifs en calcul, et de nombreuses solutions possibles à ce problème sont généralement demandées par le vérificateur **V**. Si **P** connaît des informations critiques relatives à la solution, il répond avec l'une des solutions demandées disponibles. Si P ne connaît rien des informations critiques, il est alors incapable de fournir les informations nécessaires au vérificateur **V**. [82]

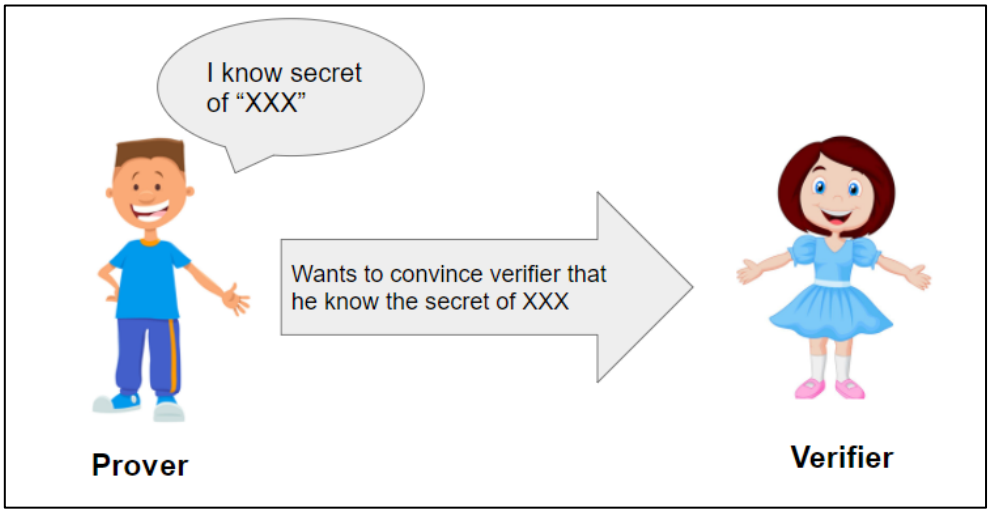

Figure 14- Concept de Zero-Knowledge Proof. [81]

4. Les principaux acteurs du ZKP

Dans le cadre du protocole ZKP, il est crucial de comprendre les acteurs clés qui interviennent au cœur de chaque interaction pour garantir la confidentialité et l'intégrité des données échangées. Ces acteurs sont :

4.1. L'utilisateur :

est l'individu qui souhaite maintenir l'accès en toute sécurité à un compte auprès d'un ou de plusieurs services.

4.2. L'appareil de confiance :

est un dispositif de confiance qui appartient à l'utilisateur et est nécessaire à chaque tentative de connexion. Il doit être un appareil mobile non compromis.

4.3. Appareil non fiable (Untrusted Device) :

est un dispositif utilisé par l'utilisateur pour se connecter à un compte, mais il n'est pas sûr et peut être compromis. Cela signifie que quelqu'un d'autre pourrait avoir accès à ce que vous tapez (keylogging) ou faire du "shoulder surfing" (espionnage). De plus, le dispositif non fiable est également susceptible d'être verrouillé par ces méthodes. En raison de ces risques, il est essentiel de prendre des précautions supplémentaires lors de l'utilisation d'un appareil non fiable pour se connecter.

4.4. Serveur :

c'est l'appareil chargé d'un service particulier. Toute interaction avec le service est réalisée grâce à la communication avec le serveur.

4.5. Adversaire :

il regroupe toutes les actions malveillantes possibles sous une seule hypothèse d'un individu fictif que nous appelons l'adversaire. [83]

5. Les Composants clés du ZKP

ZKP représente une avancée majeure en cryptographie, offrant une méthode sophistiquée pour démontrer la véracité d'une information sans la révéler. Cette technique repose sur un ensemble de composantes essentielles :

5.1. Le témoin :

Il s'agit de l'information confidentielle que le prouveur souhaite démontrer sans pour autant la révéler. Cette donnée constitue la base de la preuve et représente ce que le prouveur veut prouver, souvent sans que le vérificateur n'ait besoin de connaître les détails de cette information.

5.2. L'interrogatoire :

Cette phase implique une série de questions posées par le vérificateur dans le but de tester la connaissance du prouveur sans pour autant avoir accès au témoin. Ces questions sont conçues de

manière à mettre à l'épreuve la compréhension et la légitimité des affirmations du prouveur, sans compromettre la confidentialité du témoin.

5.3. La réponse :

C'est le processus par lequel le prouveur démontre sa connaissance du témoin sans le divulguer. Cette étape est cruciale car elle permet au prouveur de convaincre le vérificateur de la validité de ses affirmations sans compromettre la confidentialité du témoin. En d'autres termes, la réponse fournie par le prouveur doit être convaincante et crédible, démontrant ainsi sa capacité à manipuler l'information sans la révéler.[84]

6. Propriétés principales du ZKP

Afin d'explorer les propriétés fondamentales de ZKP, il est crucial de comprendre les principes sous-jacents de ces concepts cryptographiques.

Le prouveur cherche à démontrer qu'il détient certaines informations auprès du vérificateur sans révéler leur contenu. Simultanément, le vérificateur examine la preuve et décide de l'accepter ou de la rejeter. Ces schémas se caractérisent par trois propriétés distinctes qui doivent être satisfaites : [82]

6.1. Consistance (Completeness) :

Si ce que le prouveur dit est vrai, il peut toujours convaincre le vérificateur de sa véracité. Par exemple, si le prouveur dit qu'il connaît le mot de passe et il le connaît vraiment, il peut prouver cela au vérificateur.

6.2. Robustesse (Soundness) :

Si ce que le prouveur dit est faux, il ne pourra pas convaincre le vérificateur, sauf dans de très rares cas. Par exemple, si le prouveur prétend connaître le mot de passe mais il ne le connaît pas, il ne pourra presque jamais convaincre le vérificateur de la véracité de cette fausse affirmation.

6.3. Zéro Connaissance (Zero-Knowledge):

Le prouveur peut prouver qu'il connaît quelque chose sans révéler cette chose elle-même. Par exemple, il peut prouver qu'il connaît le mot de passe sans révéler le mot de passe. Il existe une méthode pour que le vérificateur soit convaincu sans apprendre le mot de passe, seulement qu'il existe un mot de passe correct.

Ces propriétés permettent d'assurer la confidentialité et la sécurité dans de nombreux systèmes cryptographiques et applications de vérification.

7. Fonctionnement d'un Système ZKP :

Le ZKP est un outil cryptographique puissant permettant de prouver des faits sans divulguer de données sensibles. Afin de mieux en comprendre le fonctionnement, des exemples seront illustrés pour fournir une explication approfondie de chaque étape de ce concept cryptographique.

7.1. La grille de Sudoku

La grille de Sudoko , exposé en figue 15, est un exemple classique de preuve à connaissance nulle, souvent utilisé pour expliquer les concepts de ZKP.

Dans cet exemple, Bob souhaite démontrer à Alice qu'il possède la solution d'une grille standard de Sudoku. Pour ce faire, une fois la grille résolue, il découpe soigneusement celle-ci en 81 petits carrés de papier, chacun portant un chiffre de la grille de Sudoku résolue. Ensuite, il retourne tous les carrés de papier, à l'exception des chiffres initiaux de la grille (ceux présents avant le début de la résolution). La procédure de preuve peut alors commencer.

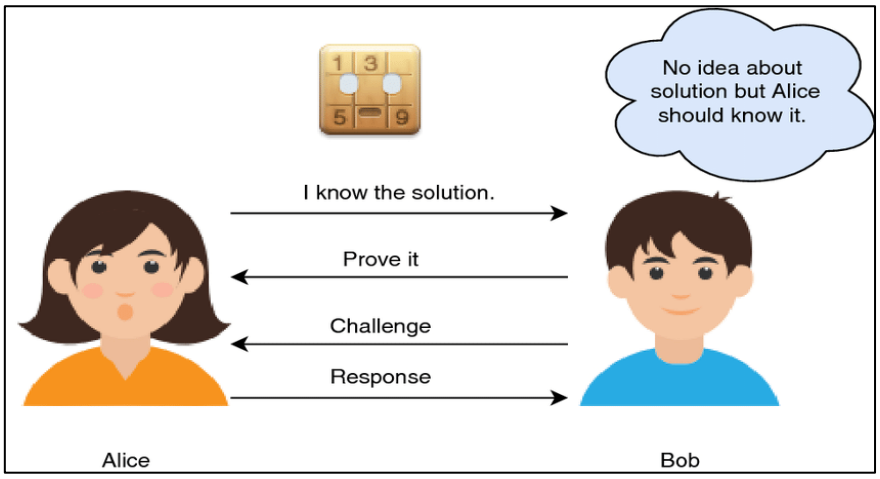

Figure 15- Principe de La grille du Sudoku. [85]

Alice choisit ensuite une ligne, une colonne ou un carré 3x3 au hasard. Bob prend alors les 9 carrés de papier correspondants, les mélange méticuleusement et les remet à Alice. Alice peut alors vérifier que les chiffres de 1 à 9 sont présents. Ensuite, Bob replace les carrés de papier à leur emplacement initial, face cachée sur la grille. Alice répète cette procédure jusqu'à ce qu'elle ait vérifié les 9 lignes, les 9 colonnes et les 9 carrés 3x3 de la grille de Sudoku , comme vous pouvez le constater dans la figure 16.

	C1	C2	C3	C4	C5	C6	C7	C8	C9
L1	8	2 6	1	3	5	7	4	2 6	9
L2	7	2 6 9	6 9	8	1	4	2 6	5	3
L3	4	5	3	2	6	9	7	1	8
L4	5 6	3	2 7	6 9	4	8	1	2 9	5 7
L5	9	4	2 7	5	2 7	1	3	8	6
L6	5 6	1	8	6 9	2 7	3	2 9	4	5 7
L7	3	6 9	4	1	8	5	6 9	7	2
L8	1	8	6 9	7	3	2	5	6 9	4
L9	2	7	5	4	9	6	8	3	1

Figure 16– La grille du Sudoku

À la fin de la procédure, Alice a la certitude que Bob possède la solution de la grille de Sudoku, sans pour autant la connaître. [86]

7.2. Problème de Coloration de Graphes à 3 Couleurs

Le système ZKP pour la coloration de graphes à 3 couleurs, comme exposé à la figure 17, permet de prouver une solution sans la révéler, en révélant uniquement des informations cruciales. Le prouveur cherche à convaincre le vérificateur de sa connaissance de la coloration à 3 couleurs d'un graphe donné sans divulguer la solution exacte. Le processus est le suivant : [87]

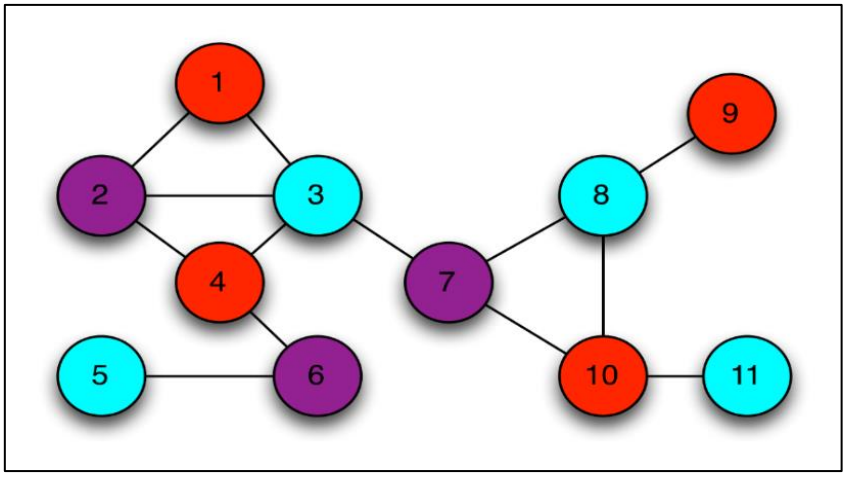

Figure 17– Coloration de graphe à 3 couleurs [76]

❖ **Masquage des Sommets :**

Le prouveur attribue une couleur à chaque sommet du graphe, formant une solution de coloration. Cependant, il masque cette coloration en mélangeant les couleurs ou en les codant d'une manière qui les rend indiscernables pour le vérificateur.

❖ **Vérification des Arêtes :**

Le vérificateur choisit aléatoirement une arête du graphe. Le prouveur révèle les couleurs des deux sommets de cette arête. En montrant que les deux sommets sont de couleurs différentes, le prouveur démontre sa connaissance de la coloration sans révéler les couleurs exactes.

❖ **Répétition du Processus :**

Ce processus est répété pour un certain nombre d'arêtes et de tours. À chaque tour, les couleurs sont mélangées de manière aléatoire, garantissant la difficulté pour le vérificateur de reconstituer la coloration exacte du graphe.

❖ **Propriétés du Protocole :**

Les propriétés essentielles du protocole, telles que la complétude (garantissant qu'un prouveur honnête peut convaincre le vérificateur), la sonorité (assurant qu'un prouveur malhonnête ne peut pas tromper le vérificateur) et la connaissance nulle (garantissant que le vérificateur n'apprend rien de la solution réelle au-delà de sa validité), sont rigoureusement démontrées.

❖ **Extension à la Classe NP-Complet :**

Étant donné que le problème de coloration de graphes à 3 couleurs est NP-complet, le protocole ZKP peut être étendu pour vérifier toute déclaration dans la classe NP, offrant ainsi une solution généralisable à un large éventail de problèmes complexes.

Ce système ouvre la voie à des avancées significatives dans la cryptographie et la sécurité informatique, avec des applications potentielles allant de la sécurisation des transactions financières aux protocoles de vote électronique.

8. Les protocoles de ZKP

Les protocoles de ZKP sont des outils fondamentaux et essentiels dans de nombreux domaines, allant de la sécurité des données à la confidentialité des transactions en ligne.

Les protocoles ZKP peuvent être classés en deux catégories principales : interactifs et non interactifs. Chacun de ces types offre des avantages uniques, adaptés à différentes situations et exigences de sécurité.

8.1. Protocoles ZKP interactifs (IZKP) :

Les protocoles interactifs ZKP impliquent une interaction entre le prouveur et le vérificateur. Dans ces protocoles, le prouveur et le vérificateur s'engagent dans un processus d'échange où le prouveur fournit des informations au vérificateur, qui les vérifie. Cela se poursuit jusqu'à ce que le vérificateur soit convaincu de la validité de l'affirmation ou qu'une limite d'itération soit atteinte, comme représenté à la figure 18. [88]

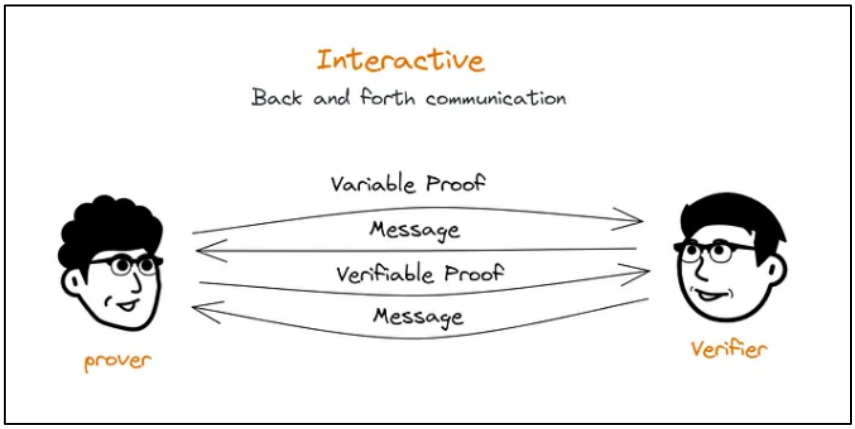

Figure 18- Protocole ZKP Interactif (IZKP). [89]

Pour mieux appréhender le fonctionnement de ce type de protocoles, considérons l'exemple ci-après.

8.1.1. Protocole de Schnorr :

Ce protocole permet de prouver la connaissance d'un logarithme discret secret (***x***) d'un élément public (***y***) tel que $y = g^X \bmod p$

Étant donné ***p,q,g*** et ***y***. il faut prouver la connaissace de ***(x).***

Dans ce protocole, le prouveur choisit un nombre aléatoire (***r***) et calcule ($t = g^r \bmod p$), où **(*g*)** et (***P***) sont convenus à l'avance (**étape d'engagement**).

Le prouveur envoie (***t***) au vérificateur, qui choisit un défi aléatoire (***c***) et l'envoie au prouveur.(**étape de Défi**).

Le prouveur calcule ensuite $(s = r - c \cdot x \bmod q)$, où (***x***) est la connaissance qu'il veut prouver, et envoie (***s***) au vérificateur (**étape de la réponse**).

Le vérificateur vérifie si ($g^s \cdot y^c = t$). Si l'équation est vérifiée, le prouveur a réussi à prouver sa connaissance sans divulguer l'information. (**étape de vérification**) [90]

Il convient de noter que ce protocole est implémenté dans notre application en raison de son équilibre idéal entre sécurité et efficacité. Ce choix s'est imposé pour sa capacité à permettre aux parties impliquées de prouver leurs connaissances sans révéler d'informations sensibles supplémentaires. Étant largement utilisé dans le domaine de la cryptographie, ce protocole assure une compatibilité étendue et une interopérabilité garantie. Dans le prochain chapitre, nous détaillerons son intégration dans notre application, en mettant en évidence ses avantages et son impact sur la sécurité des données.

8.2. Protocoles ZKP Non-interactifs (NIZKP)

Les protocoles ZKP non interactifs permettent au prouveur de générer une preuve vérifiable sans interaction supplémentaire avec le vérificateur. Dans ces protocoles, le prouveur peut générer une preuve indépendamment et la fournir au vérificateur pour vérification. Cela permet une vérification efficace sans nécessiter de communication supplémentaire, comme représenté à la figure 19.

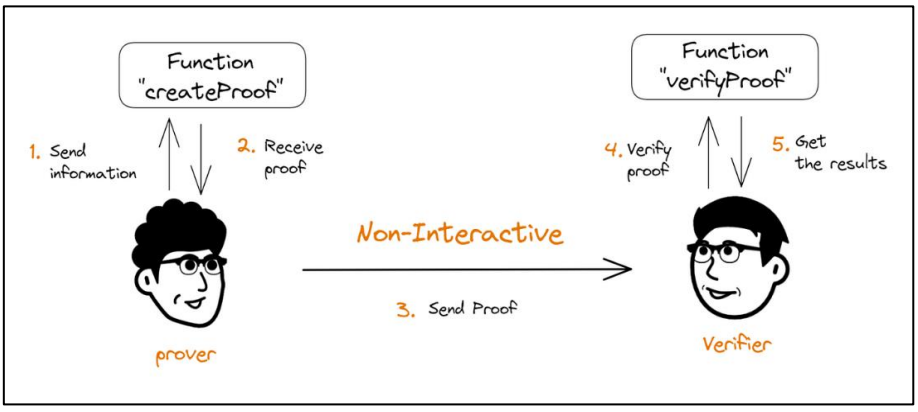

Figure 19- Protocoles ZKP Non-Interactifs (NIZKP). [89]

Pour mieux comprendre le fonctionnement de ce protocole, illustrons un exemple concret ci-dessous.

8.2.1. Protocole de Feige-Fiat-Shamir

Dans ce protocole, le prouveur génère ($A = g^X \bmod p$) , comme dans le protocole de Schnorr. Ensuite, au lieu de recevoir un défi du vérificateur, le prouveur utilise une fonction de hachage cryptographique pour générer un défi basé sur (A). Ce défi auto-généré permet au prouveur de calculer une réponse $(r = x + c \cdot w)$, où (w) est la connaissance qu'il veut prouver. Le prouveur envoie ensuite (r) au vérificateur, qui vérifie si ($A = g^r \cdot w^c \bmod p$). Ce processus rend le protocole non interactif car il élimine la nécessité d'une interaction directe entre le prouveur et le vérificateur pour générer le défi. [91]

Le protocole de Feige-Fiat-Shamir élimine l'étape interactive en utilisant un oracle aléatoire. Au lieu d'engager une interaction directe avec le vérificateur, l'entité se sert de l'oracle aléatoire pour créer des défis basés sur le secret. Ces défis sont transmis au vérificateur sous forme de preuves non interactives. L'oracle joue le rôle d'un simulateur d'interaction, agissant comme une "boîte noire" capable de fournir des réponses aléatoires à certaines requêtes, sans que les parties connaissent les réponses réelles à l'avance, préservant ainsi la confidentialité du secret. Grâce à ce processus, le protocole permet à l'entité de prouver son identité de manière non interactive.

Ces protocoles jouent un rôle crucial dans de nombreux aspects de la sécurité informatique et de la confidentialité des données, fournissant des mécanismes robustes pour garantir l'intégrité et l'authenticité des informations dans un environnement en ligne de plus en plus complexe et interconnecté.

8.2.2. Les principaux protocoles de NIZKP :

La cryptographie moderne offre des solutions pour garantir la confidentialité des données tout en permettant la vérification de leur authenticité.

Dans cette optique, divers protocoles ont été développés pour répondre à différents besoins en matière de confidentialité et d'efficacité. Parmi les plus significatifs figurent les zk-SNARKs, les zk-STARKs, et les Bulletproofs, chacun offrant des avantages uniques et adaptés à des cas d'utilisation spécifiques. [92]

Le protocole de Feige-Fiat-Shamir est souvent utilisé comme base pour construire des protocoles ZKP non interactifs, notamment dans les constructions de zk-SNARKs ,où il est employé dans la phase de génération des paramètres de configuration.

- **zk-SNARKs (Succinct Non-interactive ARgument of Knowledge):**

Les zk-SNARKs permettent de prouver qu'une déclaration est vraie sans divulguer plus d'informations que nécessaire. Ces preuves sont succinctes et non interactives, ce qui signifie qu'elles sont de taille réduite et peuvent être vérifiées rapidement sans nécessiter une communication entre le prouveur et le vérificateur. Cependant, ils exigent une configuration initiale sécurisée entre les deux

parties, ce qui pose un défi de centralisation. Malgré cela, des recherches sont en cours pour réduire la confiance requise dans ce processus.

❖ **zk-STARKs (Zero-Knowledge Scalable Transparent ARguments of Knowledge):**

Les zk-STARKs sont une alternative aux zk-SNARKs, offrant une mise en œuvre plus rapide et moins coûteuse. Contrairement aux zk-SNARKs, les zk-STARKs ne nécessitent pas de configuration initiale sécurisée car ils reposent sur une cryptographie plus symétrique via des fonctions de hachage résistantes aux collisions. Cela les rend également moins sensibles aux attaques des ordinateurs quantiques. De plus, le nombre de tours de communication entre les parties reste constant, ce qui les rend plus efficaces en termes de calcul. [93]

❖ **Bulletproofs:**

Les Bulletproofs sont des preuves non interactives à connaissance nulle qui ne nécessitent pas de configuration de confiance. Elles sont conçues pour permettre des transactions privées dans le domaine des crypto-monnaies.

8.2.3. Comparaison entre zk-STARKs, zk-SNARKs et Bulletproofs

Dans le domaine de la confidentialité et de la vérification de données sur les blockchains, zk-SNARKs, zk-STARKs et Bulletproofs représentent trois approches distinctes mais puissantes, comme démontré à la figure 20. Les zk-SNARKs offrent une confidentialité rapide et succincte mais nécessitent une dépendance initiale de confiance, tandis que les zk-STARKs sont transparents et plus rapides à vérifier, bien qu'ils puissent être plus lents à générer. Bulletproofs, quant à eux, se distinguent par leur efficacité en termes de vérification et leur absence de dépendance initiale de confiance, tout en offrant des preuves de confidentialité plus rapides à générer.

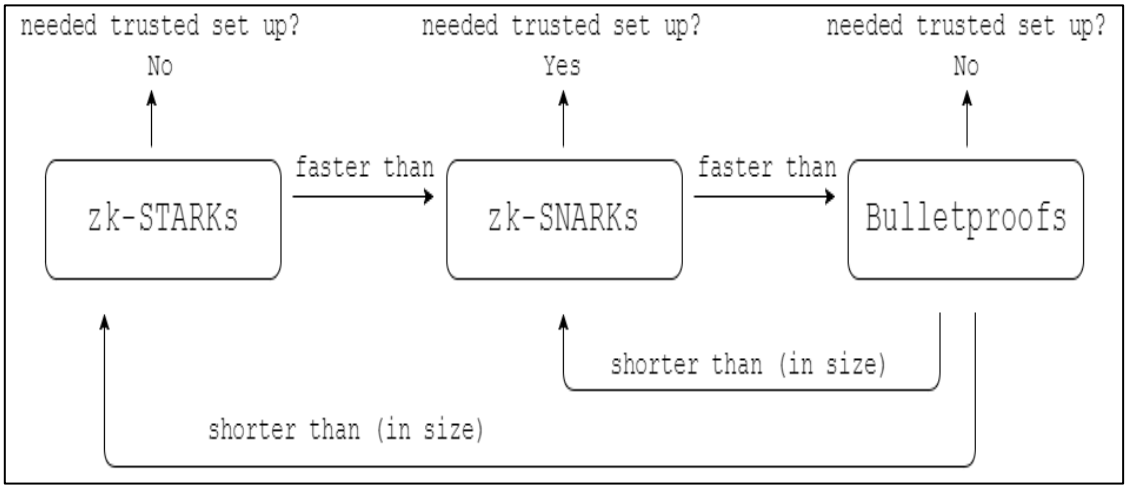

Figure 20- Comparaison entre zk-STARKs, zk-SNARKs et Bulletproofs. [94]

Chacune de ces technologies à ses propres forces et faiblesses en fonction des exigences spécifiques du cas d'utilisation, qu'il s'agisse de la vitesse de vérification, de la taille des preuves ou du niveau de confiance requis.

Ces protocoles répondent à une préoccupation croissante des utilisateurs en matière de confidentialité, notamment dans le domaine de la crypto-monnaie, et pourraient jouer un rôle important dans leur adoption par le grand public.

9. Applications de ZKP

Les protocoles ZKP ont révolutionné la sécurité informatique en proposant une méthode novatrice pour assurer la confidentialité, l'authenticité et la sécurité des données au sein des systèmes distribués. Leur intégration dans la blockchain révèle la diversité des cas d'utilisation, soulignant ainsi leur importance cruciale dans la sécurisation et la gouvernance des réseaux distribués. Parmi les applications des ZKP, on distingue notamment :

❖ Confidentialité des transactions :

Les ZKP sont amplement utilisées dans les blockchains privées et publiques pour garantir la confidentialité des transactions. Elles permettent aux utilisateurs de prouver la propriété d'informations sans les divulguer, assurant ainsi la confidentialité des données sensibles. [95]

❖ Authentification :

Les ZKP peuvent également servir à authentifier les utilisateurs dans les systèmes blockchain, offrant un moyen sécurisé et privé d'accéder aux plateformes et aux services sans révéler d'informations d’identification. [96]

❖ Vérification d'identité :

Dans les systèmes basés sur la blockchain, les ZKP peuvent être utilisées pour vérifier l'identité des utilisateurs sans divulguer d'informations personnelles, garantissant ainsi que seules les parties autorisées ont accès aux données pertinentes. [7]

❖ Audit et conformité :

Les ZKP facilitent l'audit et la conformité dans les réseaux blockchain en permettant aux parties de prouver leur conformité à des règles prédéfinies sans divulguer d'informations sensibles. [97]

❖ Gouvernance décentralisée :

Les ZKP peuvent garantir l'intégrité des processus de vote et des décisions prises par les parties prenantes dans les systèmes de gouvernance décentralisée.

Ces applications mettent en évidence l'importance des ZKP dans la sécurisation et la gestion efficace des transactions et des données au sein des réseaux blockchain. [98]

❖ Zcash :

Zcash est une cryptomonnaie qui utilise des protocoles ZKP appelés zk-SNARKs pour garantir la confidentialité des transactions. Les utilisateurs peuvent effectuer des transactions sans révéler les adresses des expéditeurs, des destinataires ou les montants transférés. [99]

10.Les développements récents et futurs de ZKP

Les développements récents et futurs de ZKP promettent d'apporter des avancées significatives dans plusieurs domaines. A savoir :

❖ Standardisation :

Des efforts sont en cours pour standardiser les protocoles ZKP, ce qui permettra une adoption plus large et une meilleure interopérabilité entre différentes implémentations. Cette standardisation facilitera également l'intégration du ZKP dans diverses applications et systèmes. Par exemple, l'initiative de la Global Standards Collaboration (GSC) examine actuellement les protocoles ZKP en vue de leur inclusion potentielle dans les normes internationales.

❖ Recherche active :

La recherche dans le domaine du ZKP est en constante évolution, visant à améliorer l'efficacité des protocoles existants et à découvrir de nouveaux protocoles plus performants. Ces efforts de recherche contribueront à rendre le ZKP plus robuste, efficace et applicable à un plus large éventail de cas d'utilisation. Par exemple, des chercheurs ont récemment proposé de nouvelles techniques d'optimisation pour réduire les coûts de calcul et de communication associés à l'utilisation du ZKP.

❖ Intégration dans les technologies émergentes :

Le ZKP est de plus en plus considéré comme une solution prometteuse pour renforcer la sécurité et la confidentialité dans des domaines émergents tels que l'intelligence artificielle, l'IoT et le Cloud Computing. Son utilisation dans ces domaines pourrait permettre de garantir la confidentialité des données sensibles, de renforcer la confiance entre les parties et de promouvoir l'adoption de ces technologies émergentes à grande échelle. Par exemple, des entreprises de technologie telles que Microsoft et IBM explorent activement l'utilisation du ZKP pour sécuriser les données dans le cloud et renforcer la confidentialité des utilisateurs.

Globalement, les développements récents et futurs du ZKP visent à renforcer sa pertinence et son utilité dans un large éventail de domaines, tout en répondant aux défis de sécurité et de confidentialité auxquels sont confrontées les technologies modernes.

11.Langages utilisés pour implémenter ZKP

Lorsqu'il s'agit d'implémenter ZKP, le choix du langage de programmation joue un rôle crucial, car différents langages offrent des avantages distincts pour ce type de développement[100]. Les principaux langages utilisés sont :

❖ Solidity :

Utilisé pour écrire des contrats intelligents sur la blockchain Ethereum, couramment employé pour les applications décentralisées (dApps) et les systèmes basés sur la blockchain.

❖ **Rust :**

Connu pour sa sécurité mémoire et ses performances, utilisé pour implémenter des bibliothèques de Preuves à Connaissance Zéro comme "bellman" de Zcash ou "arkworks".

❖ **C++ :**

Souvent utilisé pour implémenter des algorithmes cryptographiques et des protocoles complexes, incluant de nombreuses bibliothèques de Preuves à Connaissance Zéro comme "libsnark".

❖ **Python :**

Populaire pour le prototypage rapide, avec des bibliothèques telles que "pySNARK" et "Petlib" pour les implémentations de Preuves à Connaissance Zéro. Il est également utilisé pour le scripting et les applications interagissant avec ces systèmes.

❖ **JavaScript :**

Principalement utilisé pour le développement web, il peut être employé pour les Preuves à Connaissance Zéro dans des applications basées sur navigateur, avec des bibliothèques comme "ZoKrates".

❖ **Go :**

Adapté à la construction de systèmes backend, utilisé pour les Preuves à Connaissance Zéro avec des bibliothèques telles que "gnark".

❖ **Java :**

Employé pour implémenter les Preuves à Connaissance Zéro dans diverses applications, y compris les systèmes d'entreprise et les registres distribués, grâce à ses bibliothèques cryptographiques robustes.

12.Conclusion

Zero knowledge Proof (ZKP) est une technologie remarquable qui doit encore faire ses preuves. En effet, aujourd'hui le ZKP est très peu utilisé pour plusieurs raisons énoncées plus haut, notamment à cause du manque de développeurs sur ce sujet complexe. Cependant, les cas d'utilisation de cette technologie sont de plus en plus nombreux, principalement dans le domaine de la blockchain.

Le ZKP représentent une avancée majeure dans le domaine de la cryptographie, offrant un moyen puissant de sécuriser les échanges d'informations sensibles sans compromettre la confidentialité. Leur capacité à garantir l'authenticité, à faciliter l'identification et l'échange de clés tout en préservant la confidentialité des données en fait un outil essentiel dans la conception et la mise en œuvre de protocoles cryptographiques robustes. En réduisant les exigences computationnelles par rapport à d'autres protocoles, les ZKP offrent une solution efficace et polyvalente pour un large éventail d'applications, ce qui en fait un pilier fondamental de la sécurité numérique moderne.

Dans le prochain chapitre, nous illustrons une implémentation de ZKP pour l’authentification des transactions dans une Blockchain

CHAPITRE IV :

L’IMPLEMENTATION D’APPLICATION

CHAPITRE IV : IMPLEMENTATION D'APPLICATION

1. Introduction

Le Zero Knowledge proof (ZKP) est une technologie remarquable qui, malgré son potentiel, reste relativement sous-exploitée en raison de divers obstacles, tels que le manque de développeurs spécialisés. Pourtant, son utilisation est de plus en plus envisagée, notamment dans le contexte de la blockchain, où ses avantages en matière de sécurité et de confidentialité sont particulièrement précieux.

Ce chapitre se concentre sur la réalisation pratique de ce mémoire, concrétisant ainsi les concepts théoriques abordés dans les chapitres précédents.

Nous débuterons par une vue d'ensemble des outils, de l'environnement de développement et des techniques déployées pour atteindre nos objectifs. Ensuite, nous fournirons un aperçu détaillé des interfaces et des parties de code les plus significatives de notre projet, mettant notamment en avant l'utilisation de ZKP dans un contexte de blockchain.

Ensuite, nous explorerons l'architecture de notre projet, en détaillant le déploiement des différents modules qui le composent.

Enfin, nous conclurons en récapitulant les principales fonctionnalités mises en œuvre et en discutant des défis rencontrés ainsi que des perspectives pour les développements futurs de cette technologie dans le domaine de la blockchain.

2. Présentation du Projet

Notre projet se concentre sur l'implémentation d'un système d'authentification des transactions au sein d'une blockchain en utilisant le ZKP. En tirant parti de cette technique, l'objectif est de garantir une authentification et une vérification sécurisées tout en préservant la confidentialité des informations sensibles. Cette approche revêt une importance cruciale dans les environnements décentralisés où la sécurité et la confidentialité des transactions sont des enjeux majeurs.

2.1. Environnement de travail

2.1.1. Matériel physique :

Le travail a été réalisé avec un PC dont la configuration est la suivante :

- ✓ Marque ASUS, Model : VIVOBOOK 509J ,Processeur Intel (R) Core i3-1005G1, 4,00 Go de RAM et un Système d'exploitation Windows 10 pro 64bits.

2.1.2. Langage de programmation :

Nous avons choisi Python comme langage de développement pour plusieurs raisons, à savoir:

- ✓ **Flexibilité et simplicité** : Python est reconnu pour sa simplicité et sa facilité d'utilisation, ce qui accélère le développement et la mise en œuvre des solutions.
- ✓ **Richesse des bibliothèques** : Python dispose de nombreuses bibliothèques cryptographiques bien établies, telles que « cryptography », « hashlib » et « random », qui facilitent l'implémentation du notre protocole et d'autres fonctionnalités cryptographiques.
- ✓ **Grande communauté de développeurs** : La vaste communauté de développeurs Python assure un support et des ressources abondantes, ce qui est précieux pour le développement, le dépannage et l'amélioration continue de notre application.

2.1.3. Outil de développement :

Dans notre application, nous avons opté pour PyCharm comme outil de développement. PyCharm est un environnement de développement intégré (IDE) conçu spécifiquement pour Python.Il facilite la création, la modification et la gestion du code Python grâce à ses fonctionnalités intuitives et puissantes telles que la détection d'erreurs en temps réel ainsi que des capacités de navigation avancées.

2.1.4. Protocoles appliqués :

Dans le développement de notre application, nous avons implémenté les protocoles suivants :

- ✓ **ZKP :** qui permet de vérifier l'identité d'un utilisateur sans révéler l'information secrète elle-même. Cette méthode est cruciale pour maintenir la confidentialité tout en garantissant une authentification sécurisée. Pour notre application, l'utilisation du protocole de Schnorr dans le contexte d'authentification repose sur plusieurs considérations, notamment, une sécurité robuste, une efficacité en termes de calcul et une meilleure confidentialité par rapport à d'autres protocoles. En particulier, Schnorr permet l'agrégation des signatures, ce qui réduit la taille des transactions

et améliore l'efficacité de la blockchain.

- ✓ **POW (Proof of Work):** Le consensus est le mécanisme par lequel les nœuds de la blockchain s'accordent sur un ensemble unique de transactions vérifiées. Nous avons opté pour le consensus Proof of Work (PoW) en raison de sa robustesse et de sa résistance aux attaques. PoW est un mécanisme éprouvé qui assure que les transactions sont validées et ajoutées à la blockchain de manière sécurisée. La combinaison du protocole de Schnorr avec le consensus PoW renforce la sécurité du système en garantissant que les transactions signées sont valides et non falsifiables.

3. Fonctionnement de l'application

Pour mieux l'expliquer, nous pouvons schématiser le processus de fonctionnement sous forme de diagramme dans la figure 21et 22:

- ➢ **Partie blockchain :** Cette phase est constituée des étapes suivantes :

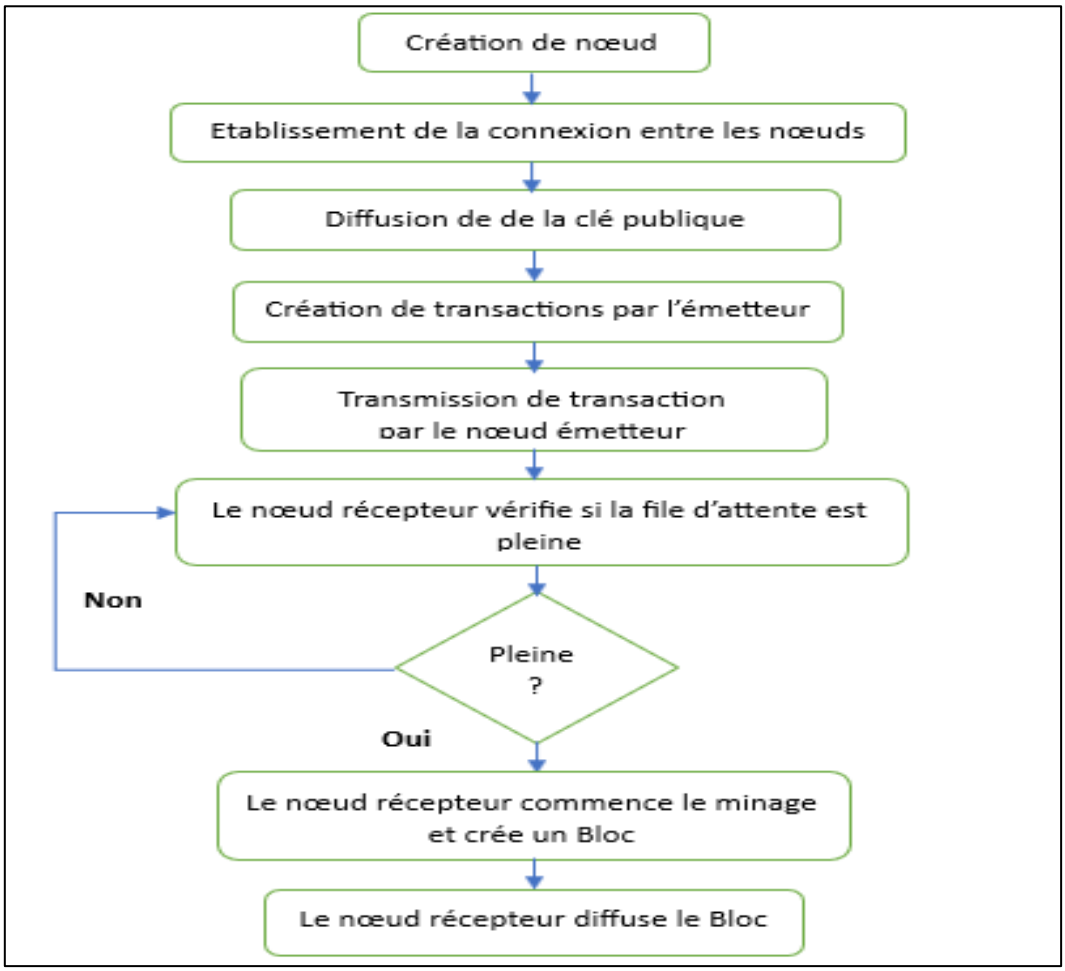

Figure 21- Diagramme de création de Blockchain.

- ➢ A**uthentification à l'aide de ZKP :** Le ZKP permet de vérifier et valider les transactions dans la Blockchain de manière sécurisée sans révéler l'information privée. La figure 22 représente le diagramme des étapes de l'authentification ZKP :

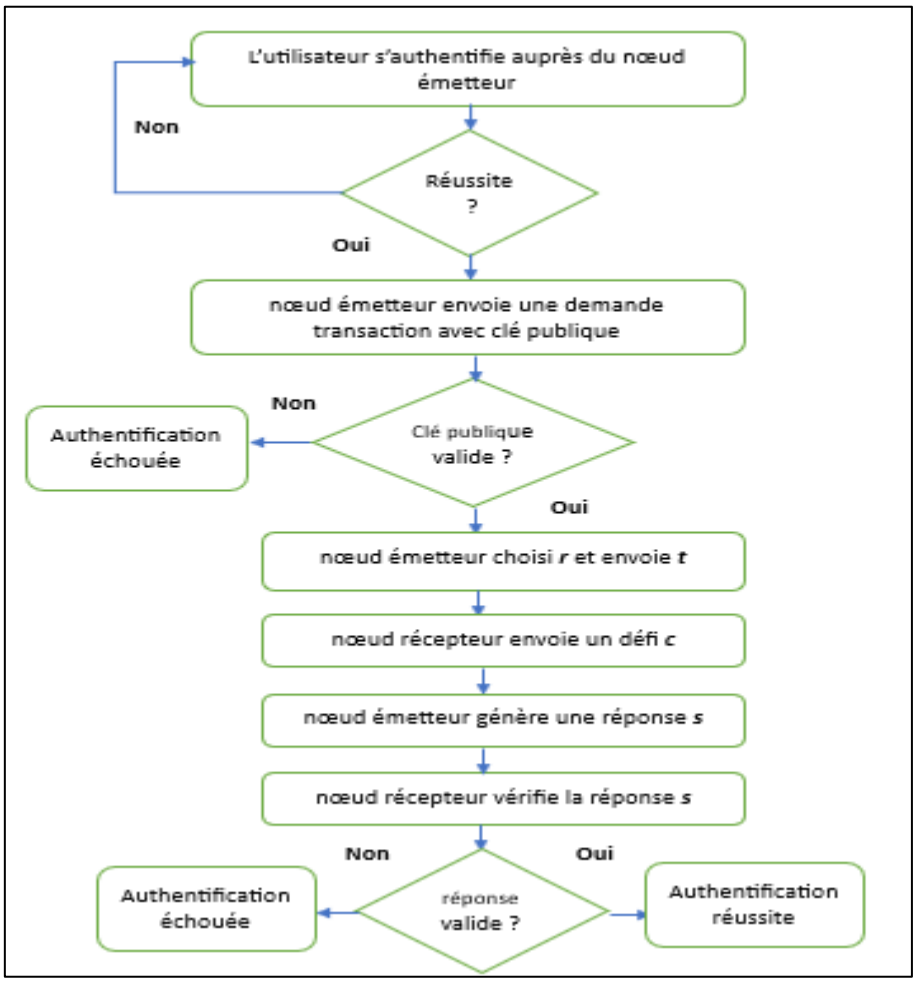

Figure 22- Diagramme de fonctionnement de ZKP

Voici les étapes clés de fonctionnement de ZKP :

- **Génération de Preuves** : Le prouveur utilise ses connaissances pour générer une preuve qui sera vérifiée par le vérificateur.
- **Calcul du Challenge**: Le vérificateur génère un challenge aléatoire ou basé sur une fonction cryptographique pour tester la connaissance du prouveur.
- **Vérification des Preuves**: Le vérificateur utilise le challenge et la preuve fournie par le prouveur pour vérifier que la preuve est correcte sans avoir besoin de connaître les détails de l'information privée.
- **Intégration dans la Blockchain**: Une fois vérifiée, la transaction ou l'authentification sécurisée est intégrée dans la Blockchain, assurant ainsi la sécurité et l'intégrité des données
- **Test et validation :** Dans une blockchain, le risque d'intrusion survient lorsque des nœuds malveillants tentent de se faire passer pour des nœuds légitimes afin d'insérer des transactions non autorisées. Pour contrer cela, notre code inclut des mécanismes avancés d'authentification des transactions via ZKP. Nous avons réalisé des tests pour évaluer l'efficacité de notre application d'authentification avec ZKP. Ces tests se concentrent sur la vérification de la sécurité, de la performance et de la robustesse de l'implémentation.

Ces phases seront détaillées en se référant aux fonctions et méthodes du code illustrées en bas.

4. Structure du code implémenté

Pour mieux comprendre l'implémentation technique de notre programme, nous détaillons ci-dessous les principales étapes et composants.

4.1. Importation des modules

Les modules sont regroupés en deux catégories principales : ceux dédiés à l'affichage de l'interface utilisateur et ceux utilisés pour la gestion de la blockchain ainsi que l'authentification à l'aide de ZKP.

Les modules pour l'interface utilisateur comprennent principalement :

- ✓ « tkinter » : bibliothèque pour la création de l'interface graphique.
- ✓ « PIL » et « ImageTk » : permettent la manipulation des images dans l'interface.
- ✓ « customtkinter » :peut être utilisé pour des fonctionnalités spécifiques personnalisées.
- ✓ « ttk » et « messagebox » : fournissent des widgets stylisés et des boîtes de dialogue.

Les modules utilisés pour la gestion de la blockchain et l'authentification à l'aide de ZKP sont :

« time » et « datetime » sont utilisés pour la gestion du temps dans la blockchain.

- ✓ « random », « hashlib », et « secrets » sont employés pour la génération de nombres aléatoires, le hachage et la génération de nombres cryptographiques sécurisés respectivement, ces derniers sont nécessaires pour l'authentification à l'aide de ZKP.

Code Python

```
from tkinter import *
from tkinter import ttk, messagebox
import tkinter as tk
import customtkinter
from PIL import ImageTk, Image

from datetime import datetime
import time
import random
import hashlib
import secrets
```

Cette organisation des modules permet de séparer clairement les fonctionnalités liées à l'interface utilisateur de celles associées à la gestion de la blockchain et à l'authentification, facilitant ainsi la compréhension et la maintenance du code.

4.2. Fonctions utilitaires

Chaque ligne et chaque fonction ont des rôles spécifiques dans la création et le fonctionnement de notre application. Nous détaillons ci-dessous les principales étapes et composants.

4.2.1. Création de la Blockchain

La création et la gestion d'une blockchain contiennent plusieurs étapes clés :

❖ Class BlockchainNode:

Cette classe définit la structure pour créer un nœud au sein du réseau blockchain, incluant la création de blocs, la validation des transactions et la vérification des clés publiques.

Code Python

```
class BlockchainNode:
    def __init__(self, node_id, p, q, g):
        self.node_id = node_id
        self.blockchain = []
        self.peers = []
        self.transaction_queue = []
        self.p = p
        self.q = q
        self.g = g
        self.x = self.generate_private_key()
        self.y = self.generate_public_key(p, g, self.x)
        self.public_keys = {self.node_id: self.y}
```

❖ Calcul du hachage :

« calculate_hash »: Cette fonction Cette méthode prend en entrée les éléments d'un bloc (index, hachage précédent, données et nonce), puis calcule le hachage SHA-256 correspondant.

Code Python

```
def calculate_hash(index, previous_hash, data, nonce):
    value = str(index) + str(previous_hash) + str(data) + str(nonce)
    return hashlib.sha256(value.encode()).hexdigest()
```

❖ Initialisation d'un nœud blockchain :

« __init__ » de la classe « BlockchainNode »: Ce constructeur initialise un nouveau nœud de la blockchain avec son identifiant, sa blockchain initiale (contenant uniquement le bloc de genèse), ses pairs, sa file de transactions, ainsi que les paramètres nécessaires à l'authentification ZKP.

Code Python

```
def __init__(self, node_id, p, q, g):
     self.node_id = node_id
     self.blockchain = [create_new_block(0, '0', 'Genesis Block', 0)]
     self.peers = []
     self.transaction_queue = []
     self.p = p
     self.q = q
     self.g = g
     self.x = self.generate_private_key()
     self.y = self.generate_public_key(p, g, self.x)
     self.public_keys = {self.node_id: self.y}  # Initialize with own public key
```

❖ Ajout d'un pair :

« add_peer »: Cette méthode ajoute un pair au nœud actuel pour permettre la communication entre les nœuds de la blockchain.

Code Python

```
def add_peer(self, peer):
    self.peers.append(peer)
```

❖ Diffusion d'un bloc :

« broadcast_block »: Cette méthode permet à un nœud de diffuser un bloc nouvellement crée à tous ses pairs.

Code Python

```
def broadcast_block(self, block):
    for peer in self.peers:
        peer.receive_block(block)
```

❖ Réception d'un bloc :

« receive_block »: Cette méthode permet au nœud de recevoir un bloc diffusé par un pair et de l'ajouter à sa propre chaîne de blocs (blockchain locale) s'il est valide.

Code Python

```
def receive_block(self, block):
    if self.is_valid_block(block):
            self.blockchain.append(block)
    else:
        print("Received invalid block. Discarding...")
```

❖ Validation d'un bloc :

« is_valid_block »: Cette méthode vérifie si un bloc donné est valide en vérifiant si son hachage commence par un certain nombre de zéros.

Code Python

```
def is_valid_block(self, block):
    return block['hash'].startswith("0000")
```

❖ Création et Minage d'un bloc :

« mine_block »: Cette méthode effectue la création et le minage d'un bloc en essayant différentes valeurs de nonce jusqu'à ce qu'un hachage valide soit trouvé.

Code Python

```
def mine_block(self, index, previous_hash, data):
        nonce = 0
        if len(self.blockchain) == 0:
            previous_hash = '0'
        while True:
            hash_value = calculate_hash(index, previous_hash, data, nonce)
            if hash_value.startswith("0000"):
                return {
                    'index': index,
                    'previous_hash': previous_hash,
                    'data': data,
                    'nonce': nonce,
                    'hash': hash_value
                }
            nonce += 1
```

❖ Ajout d'une transaction à la file d'attente :

« add_transaction_to_queue »: Cette méthode ajoute une transaction à la file d'attente du nœud en vue de l'inclure dans un bloc ultérieur.

Code Python

```
def add_transaction_to_queue(self, transaction):
    self.transaction_queue.append(transaction)
```

❖ Création et diffusion d'un bloc :

« create_and_broadcast_block »: Cette méthode crée un nouveau bloc contenant les transactions en attente et le diffuse à tous les autres pairs du nœud.

Code Python

```
    def create_and_broadcast_block(self, num_transaction):
        if len(self.transaction_queue) >= 3 or num_transaction == 0:
            data = self.transaction_queue.copy()
            self.transaction_queue.clear()
            if len(self.blockchain) > 0:
                previous_block = self.blockchain[-1]
            else:
                previous_hash = '0'
                previous_index = 0
                previous_nonce = 0
                previous_block    =    self.mine_block(previous_index,    previous_hash,
previous_nonce)
```

```
            new_block      =      self.mine_block(previous_block['index']      +      1,
previous_block['hash'], data)
            self.broadcast_block(new_block)
            for node_info in nodes_info:
                if node_info['node_id'] == self.node_id:
                    node_info['blockchain'].append(new_block)
```

❖ Réception et traitement d'une transaction :

« receive_transaction » : Cette méthode permet au nœud de recevoir une transaction d'un autre nœud et de l'ajouter à sa file d'attente de transactions.

Code Python

```
def receive_transaction(self, sender_id, recipient_id, amount, transaction,
num_transactions):
        current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        print(f"At {current_time}: Received valid transaction from Node {sender_id} to
Node {recipient_id} with amount {amount}. Adding to transaction queue.")
        self.add_transaction_to_queue(transaction)
        self.create_and_broadcast_block(num_transactions)
```

❖ Diffusion de la clé publique :

« broadcast_public_key » : Cette méthode permet de diffuser la clé publique du nœud à tous les autres pairs.

Code Python

```
def broadcast_public_key(self):
    for peer in self.peers:
        peer.receive_public_key(self.node_id, self.y)
```

❖ Réception et enregistrement de la clé publique :

« receive_public_key » : Cette méthode permet au nœud de recevoir la clé publique d'un autre nœud et de l'enregistrer localement.

Code Python

```
def receive_public_key(self, node_id, public_key):
    self.public_keys[node_id] = public_key
```

❖ Vérification de la présence de la clé publique :

« is_public_key_present » : Cette méthode vérifie si une clé publique donnée est déjà présente dans la liste des clés publiques enregistrées localement.

Code Python

```
def is_public_key_present(self, public_key):
    return public_key in self.public_keys.values()
```

4.2.2. Authentification ZKP :

❖ Engagement du l'émetteur :

« sender_engage »: Cette méthode simule l'engagement du l'émetteur dans le protocole ZKP en calculant et retournant un engagement et un secret.

Code Python

```
def sender_engage(self, p, g, q):
    r = random.randint(1, q)
    t = pow(g, r, p)
    return t, r
```

❖ Défi du récepteur :

« receiver_send_challenge »: Cette méthode génère et renvoie par le nœud destinataire un défi pour le récepteur dans le protocole ZKP.

Code Python

```
def receiver_send_challenge(self, q):
    return random.randint(1, q)
```

❖ Réponse du l'émetteur :

« sender_respond »: Cette méthode calcule et renvoie la réponse du l'émetteur au défi du récepteur dans le protocole ZKP.

Code Python

```
def sender_respond(self, q, x, r, c):
    s = (r - c * x) % q
    return s
```

❖ Vérification du récepteur :

« receiver_verify »: Cette méthode vérifie si la réponse du l'émetteur est valide dans le protocole ZKP.

Code Python

```
def receiver_verify(self, p, g, t, y, c, s):
     left_side = (pow(g, s, p) * pow(y, c, p)) % p
     return left_side == t
```

❖ Génération de nœuds et de transactions :

Cette fonction est responsable de deux tâches principales : l'initialisation du réseau de nœuds blockchain et l'établissement de liens entre eux.

➢ Initialisation du réseau de nœuds blockchain

Tout d'abord, elle génère une liste de nœuds avec des paramètres spécifiques définis dans `param_list`.

Code Python

```
global nodes_info
nodes_info = []
nodes = []
param_list = [{"p": 23, "q": 11, "g": 2}]
```

- ✓ *nodes_info*: Une liste globale qui stocke les informations de tous les nœuds.
- ✓ *nodes*: Une liste qui contiendra les instances de BlockchainNode.
- ✓ *param_list*: Contient les paramètres nécessaires pour la génération des clés.

Ensuite, elle crée des informations sur chaque nœud, telles que l'identifiant du nœud, sa blockchain initiale et une file d'attente de transactions vide.

Code Python

```
for i in range(num_nodes):
    params_index = i % len(param_list)
    node_params = param_list[params_index]
    node = BlockchainNode(node_id=i + 1, **node_params)
    nodes.append(node)

for node in nodes:
    node_info    =    {'node_id':    node.node_id,    'blockchain':    node.blockchain,
'transaction_queue': [] }
    nodes_info.append(node_info)
```

- ✓ *Création des nœuds*: La boucle for crée chaque nœud en utilisant les paramètres dans param_list.
- ✓ *Stockage des informations des nœuds*: Pour chaque nœud, un dictionnaire contenant son identifiant, sa blockchain, et une file d'attente de transactions vide est ajouté à nodes_info.

- **Établissement de liens entre les nœuds**

Enfin, elle établit des liens entre les nœuds en les ajoutant mutuellement comme pairs, assurant ainsi une connectivité au sein du réseau.

Code Python

```
for node in nodes:
    other_nodes = [n for n in nodes if n != node]
    for other_node in other_nodes:
        node.add_peer(other_node)
```

Chaque nœud est connecté à tous les autres nœuds comme pairs, assurant une communication complète au sein du réseau.

- **Diffusion des clés publiques**

Cette partie du code commence par diffuser les clés publiques de tous les nœuds participants.

Code Python

```
for node in nodes:
    node.broadcast_public_key()
```

Chaque nœud diffuse sa clé publique pour permettre aux autres nœuds de l'utiliser lors des transactions.

- **Simulation des transactions**

Ensuite, il simule un processus de transaction. Pour chaque transaction générée aléatoirement, il vérifie l'authenticité des nœuds émetteur et destinataire en utilisant le protocole ZKP.

Code Python

```
remaining_transactions = num_transactions
while remaining_transactions > 0:
    sender_id, recipient_id, amount = generate_random_transaction(nodes)
    sender_node = next((node for node in nodes if node.node_id == sender_id), None)
    recipient_node = next((node for node in nodes if node.node_id == recipient_id), None)

    if sender_node and recipient_node:
        if recipient_node.is_public_key_present(sender_node.y):
            t,     r     =     sender_node.sender_engage(sender_node.p,     sender_node.g,
sender_node.q)
            c = recipient_node.receiver_send_challenge(sender_node.q)
            s = sender_node.sender_respond(sender_node.q, sender_node.x, r, c)
            if     recipient_node.receiver_verify(sender_node.p,     sender_node.g,     t,
sender_node.y, c, s):
```

```
                transaction = {'sender': sender_id, 'recipient': recipient_id, 'amount': amount}
                recipient_node.receive_transaction(sender_id, recipient_id, amount, transaction,remaining_transactions)
                remaining_transactions -= 1
            else:
                print(f"Transaction from Node {sender_id} to Node {recipient_id} failed authentication.")
        else:
            print(f"Transaction from Node {sender_id} rejected due to invalid public key.")
```

La génération des transactions se fait dans une boucle « while » qui continue jusqu'à ce que toutes les transactions soient créées. Pour chaque transaction, un émetteur et un destinataire sont choisis aléatoirement parmi les nœuds. La clé publique de l'émetteur est vérifiée par le destinataire. Le protocole ZKP est utilisé pour valider la transaction, comprenant l'engagement de l'émetteur, l'envoi d'un défi par le destinataire, et la réponse de l'émetteur. Si la vérification réussit, la transaction est ajoutée à la blockchain du destinataire. Sinon, elle est rejetée.

❖ Génération de nœuds similaires :

« generate_similar_public_key_node »:Ce code définit une fonction qui crée un nouveau nœud avec des paramètres similaires à un nœud existant dans un réseau de blockchain. Il extrait les paramètres clés (`p`, `q`, `g`) du nœud existant, puis crée un nouveau nœud avec ces paramètres et la même clé publique (`y`). Le nouveau nœud est ajouté à la liste des nœuds du réseau, et la fonction renvoie ce nouveau nœud créé.

Code Python

```
def generate_similar_public_key_node(existing_node):
    p = existing_node.p
    q = existing_node.q
    g = existing_node.g
    similar_node = BlockchainNode(node_id=len(nodes_info) + 1, p=p, q=q, g=g)
    similar_node.y = existing_node.y
    nodes_info.append({
        'node_id': similar_node.node_id,
        'blockchain': similar_node.blockchain,
        'transaction_queue': []
    })
    return similar_node
```

❖ Fonction de Test d'intrusion de transactions:

La fonction « `test_intrusion_transaction` » est conçu pour tester la robustesse de notre réseau blockchain contre des tentatives de fraude, en simulant des transactions entre des nœuds du réseau pour valider l'authenticité de notre programme , notamment l'intrusion d'un nœud malveillant. Voici une explication détaillée de chaque partie du programme :

- **Initialisation et diffusion des clés publiques :**

Code Python

```
for node in nodes:
    node.broadcast_public_key()
```

Cette boucle initialise chaque nœud du réseau en diffusant sa clé publique. Chaque nœud exécute la méthode `broadcast_public_key`, qui envoie sa clé publique aux autres nœuds du réseau, permettant ainsi à chaque nœud de connaître la clé publique des autres.

- **Génération des transactions :**

Code Python

```
remaining_transactions = num_transactions
while remaining_transactions > 0:
    sender_id, recipient_id, amount = generate_random_transaction(nodes)
```

Dans cette partie, le programme génère des transactions de manière aléatoire en utilisant la fonction `generate_random_transaction.` `num_transactions` spécifie le nombre total de transactions à générer. `generate_random_transaction` sélectionne aléatoirement un émetteur (sender_id) et un destinataire (recipient_id) parmi les nœuds du réseau (nodes), en assurant que l'émetteur et le destinataire sont différents et que les transactions ne sont pas envoyées à eux-mêmes.

- **Validation des conditions de transaction :**

Code Python

```
while recipient_id == sender_id or (sender_id == len(nodes) and recipient_id == 1) or
(recipient_id == len(nodes) and sender_id == 1):
    recipient_id = random.randint(1, len(nodes))
```

Cette boucle s'assure que l'émetteur et le destinataire choisis ne sont pas identiques et qu'ils ne sont pas situés aux extrémités du réseau, ce qui pourrait causer des anomalies dans les transactions.

- **Vérification de l'authenticité et traitement des transactions :**

Cette partie est cruciale car elle vérifie l'authenticité des transactions en présence d'un nœud malveillant, en utilisant un protocole de preuve à connaissance nulle (ZKP) entre le nœud émetteur et le destinataire.

Code Python

```
sender_node = next((node for node in nodes if node.node_id == sender_id), None)
recipient_node = next((node for node in nodes if node.node_id == recipient_id), None)

if sender_node and recipient_node:
    if recipient_node.is_public_key_present(sender_node.y):
        t, r = sender_node.sender_engage(sender_node.p, sender_node.g, sender_node.q)
        c = recipient_node.receiver_send_challenge(sender_node.q)
        s = sender_node.sender_respond(sender_node.q, sender_node.x, r, c)
        if     recipient_node.receiver_verify(sender_node.p,     sender_node.g,     t,
sender_node.y, c, s):
            transaction  =  {'sender':  sender_node.node_id,  'recipient':  recipient_id,
'amount': amount}
            recipient_node.receive_transaction(sender_node.node_id,       recipient_id,
amount, transaction, remaining_transactions)
            print(f"Transaction from Node {sender_node.node_id} to Node {recipient_id}
succeeded.")
        else:
            print(f"Transaction from Node {sender_node.node_id} to Node {recipient_id}
failed authentication.")
    else:
        print(f"Transaction  from  Node  {sender_node.node_id}  to  Node  {recipient_id}
rejected due to invalid public key.")
```

Le processus comprend les étapes suivantes :

- `sender_node.sender_engage`**:** Le nœud émetteur commence le ZKP en calculant les valeurs ***t*** et *r*, ce qui représente une tentative de prouver qu'il possède la clé privée correspondant à sa clé publique.
- `recipient_node.receiver_send_challenge`: Le nœud destinataire, se méfiant de l'émetteur potentiellement malveillant, envoie un défi c au nœud émetteur.
- `sender_node.sender_respond` : Malgré sa nature malveillante, le nœud émetteur répond au défi c avec la valeur s, cherchant à contourner la détection.
- `recipient_node.receiver_verify`: Le nœud destinataire vérifie la réponse s du nœud émetteur en utilisant les paramètres t, r, c, ainsi que les clés publiques et privées appropriées. Cette étape vise à détecter toute tentative de fraude ou d'utilisation non autorisée de clés.

Si la vérification réussit malgré les tentatives du nœud malveillant, la transaction est validée et traitée par le nœud destinataire. En cas d'échec de la vérification, un message d'échec d'authentification est affiché, signalant que la transaction a été rejetée en raison d'une clé publique non reconnue ou d'une preuve ZKP invalide, renforçant ainsi la sécurité contre les attaques potentielles.

4.2.3. Affichage des Informations sur les nœuds et les blocs

- ✓ **`display_nodes_info(nodes_info)`**: Affiche les informations sur les nœuds et les blocs dans une fenêtre dédiée.

Code Python

```
def display_nodes_info(nodes_info):
    info_window = tk.Tk()
    info_window.title("Nodes Information")
    for node_info in nodes_info:
        label = tk.Label(info_window, text=node_info)
        label.pack()
    info_window.mainloop()
```

4.2.4. L'interface graphique

Notre interface graphique principale, représenté dans la figure 23, offre aux utilisateurs un point d'entrée convivial dans l'application. Elle est conçue avec soin, intégrant des éléments tels que des labels, des boutons et des champs de texte pour une expérience utilisateur optimale. Voici ses caractéristiques clés:

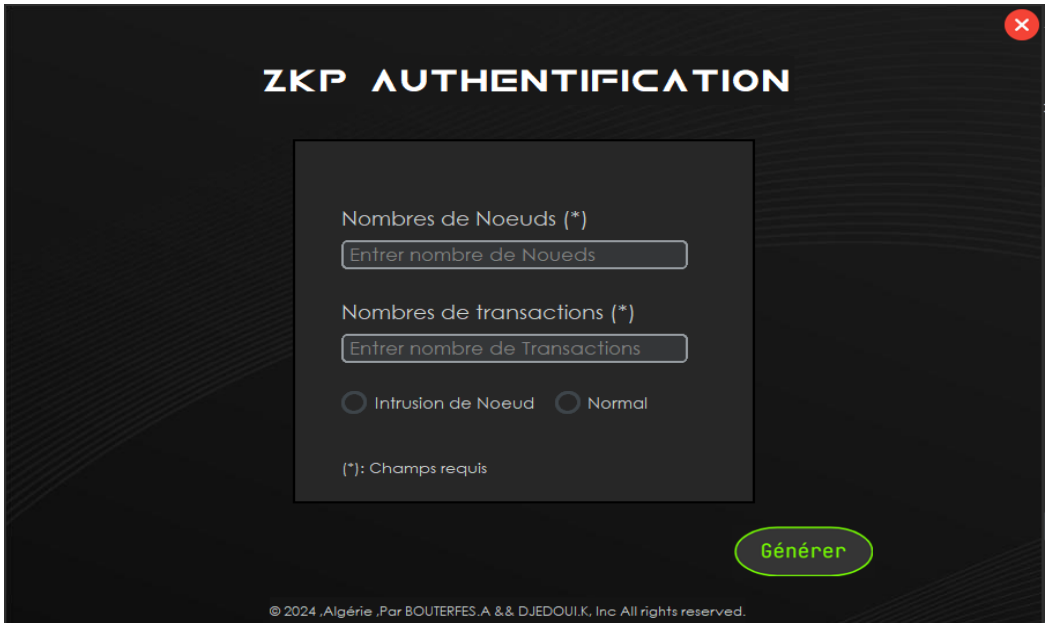

Figure 23– Interface graphique de l'application

- Interface utilisateur claire, intuitive et facile à naviguer, accompagnée d'instructions détaillées pour guider les utilisateurs dans l'utilisation de l'application.
- Bouton « Générer » permettant de démarrer le processus en un seul clic.
- Labels accompagnés de champs à remplir pour l'authentification :
  - **Nombre de nœuds** : permet aux utilisateurs de spécifier le nombre de nœuds requis.
  - **Nombre de transactions** : permet aux utilisateurs de définir le nombre de transactions nécessaires.
- Checkbox offrant deux options :
  - **Mode normal** : pour générer des blocs de transactions de manière standard.
  - **Mode Détection d'intrusion de nœuds** : permet de vérifier toute tentative d'intrusion de la part de nœuds malveillants se faisant passer pour des nœuds légitimes.

Ces fonctionnalités combinées offrent aux utilisateurs un contrôle précis sur le processus, tout en garantissant une sécurité et une efficacité optimales.

❖ Fonction de Création de la Fenêtre de Chargement

✓ **`create_loading_window()`**: Crée une fenêtre principale de l'interface ainsi qu'une fenêtre de chargement avec une animation pendant que le programme se charge.

Code Python

```
def create_loading_window():
    loading_window = tk.Tk()
    loading_window.title("Loading")
    loading_window.geometry("200x100")
    label = tk.Label(loading_window, text="Loading...")
    label.pack()
    loading_window.mainloop()
```

❖ Fonctions d'Interaction

✓ **`on_label_click(event)`** et **`on_label_click2(event)`**: Gèrent les clics sur les labels pour générer des transactions ou fermer la fenêtre.

Code Python

```
def on_label_click(event):
    on_generate_button_click()
def on_label_click2(event):
    root.destroy()
```

Dans le programme principal, les fonctionnalités et classes exposées ci-dessus sont mises en œuvre pour établir et contrôler la blockchain, initier des transactions, et les valider via ZKP, en plus d'afficher des informations utiles dans l'interface graphique. Ces aspects seront démontrés à travers un exemple concret dans la section suivante.

5. Exemple d'utilisation

Nous allons illustrer notre application dans un cas d'utilisation spécifique. Cet exemple met en lumière l'interaction entre la création de la blockchain et les processus d'authentification, démontrant ainsi l'efficacité et la pertinence de ZKP.

5.1.1. Configuration des Paramètres:

L'utilisateur définit le nombre de nœuds et spécifie le nombre de transactions souhaitées.

5.1.2. Lancement du Processus :

Après avoir choisi le mode normal, l'utilisateur clique sur le bouton "Générer" pour démarrer le processus de génération de transactions et de blocs, comme illustré dans l'image ci-dessous.

5.1.3. Authentification de nœud:

En cliquant sur le bouton, une nouvelle fenêtre apparaîtra pour permettre à l'utilisateur d'authentifier le nœud de son choix en insérant le mot de passe correspondant, comme illustré dans la figue 24. La génération des mots de passe est effectuée selon la formule « ID du nœud + azerty ».

Si le mot de passe est correct, le processus de création des blocs commencera. Sinon, l'accès sera refusé et le processus sera rejeté. Une notification sera alors affichée.

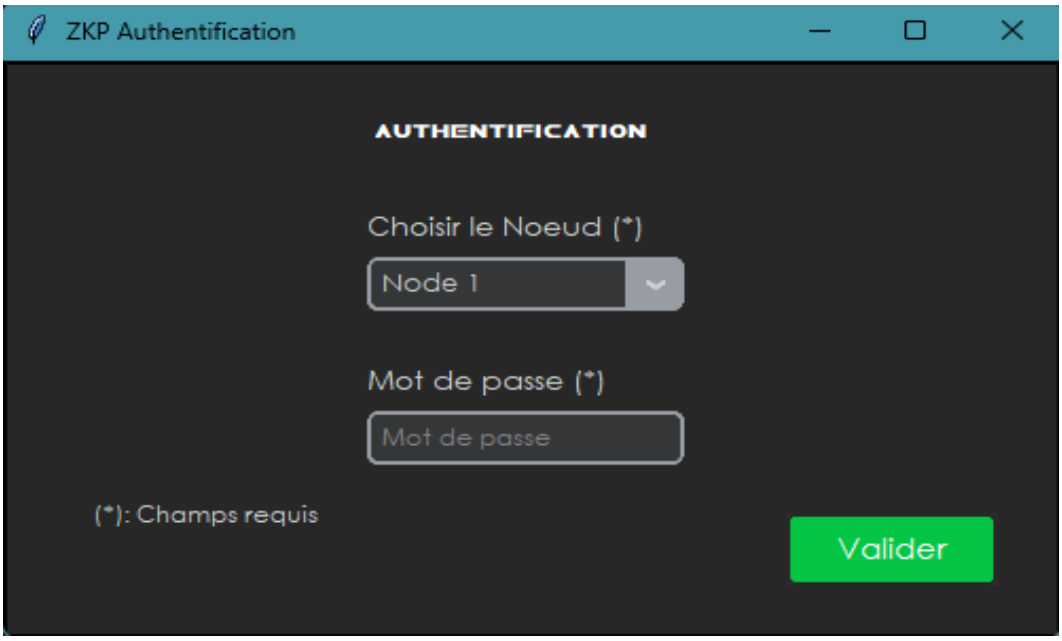

Figure 24- Authentification de nœud.

5.1.4. Création des Blocs :

Après avoir cliqué sur "Générer", l'utilisateur peut observer la création des blocs, visualisée dans l'image 25.

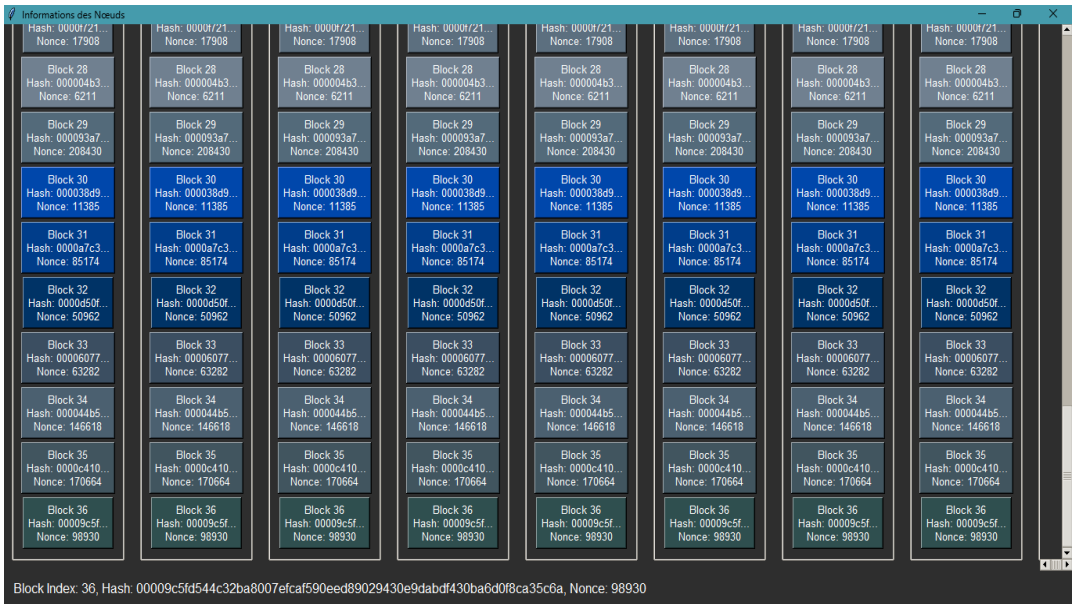

Figure 25- Blockchain générée.

5.1.5. Visualisation des transactions :

En cliquant sur l'un des blocs générés, l'utilisateur peut voir les informations détaillées de la transaction associée à ce bloc. Ces informations comprennent le temps de la transaction, le nombre total de transactions incluses dans le bloc, le montant total des transactions effectuées, ainsi que d'autres détails pertinents pour la transaction, comme mentionné dans la figure 26.

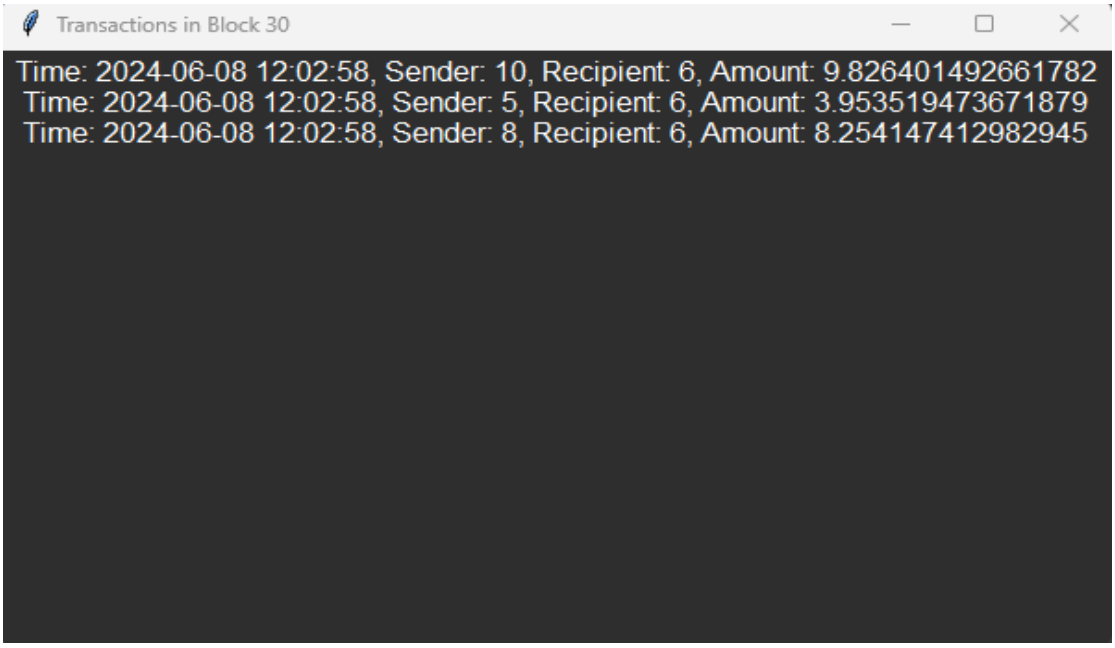

Figure 26- Visualisation de détails de transaction d'un Blocs.

5.1.6. Test d'intrusion

Dans notre exemple, illustré dans la figure 27, nous allons tester l'intrusion d'un nœud en simulant une tentative de fraude dans la blockchain. Notre application est conçue pour garantir l'intégrité du système en confirmant qu'il n'y a pas d'intrusion d'un nœud malveillant. En cas de détection d'une telle tentative, l'application affiche dans la console une notification d'erreur avec le message "Échec d'authentification du nœud". Le nœud intrus ne pourra pas inclure de bloc de transaction, comme illustré dans la figure ci-après.

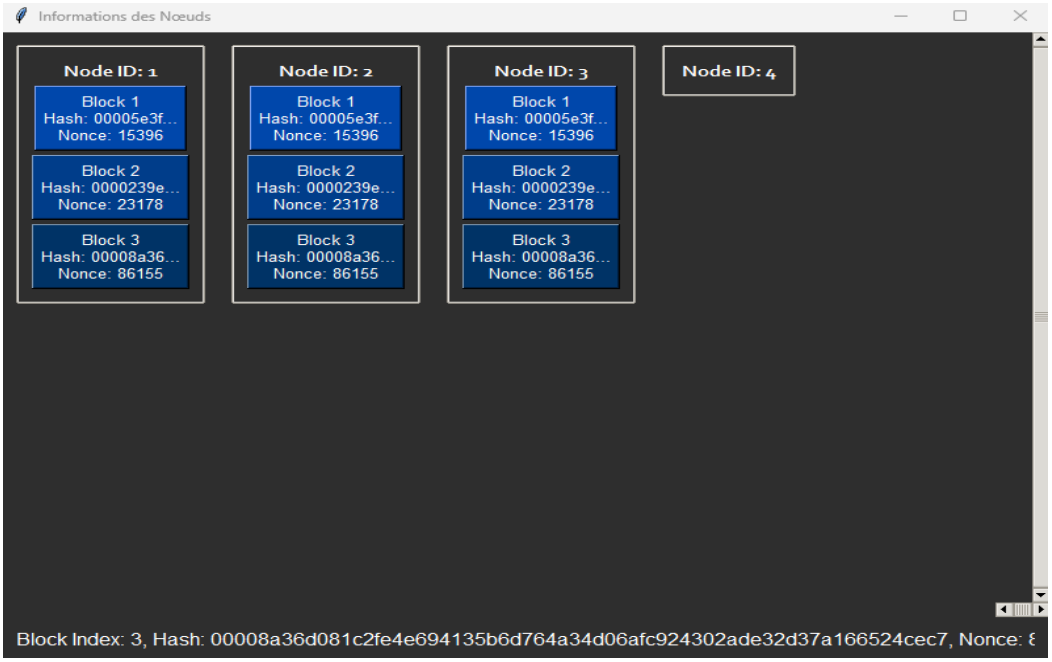

Figure 27- Visualisation de Test d'intrusion de nœuds malveillant.

Lors de la génération des transactions, le processus commence par la sélection aléatoire d'un destinataire différent de l'émetteur. Ensuite, la clé publique de l'émetteur est validée par le destinataire. Si elle est reconnue, un protocole ZKP est initié entre les parties pour prouver la possession de la clé privée correspondante. Ce protocole inclut plusieurs étapes de défi et de réponse. Si la vérification réussit, la transaction est ajoutée ; sinon, une erreur d'authentification est affichée.

En cas de tentative d'intrusion, par exemple lorsqu'un nœud malveillant utilise une clé publique similaire à celle d'un nœud légitime, le destinataire vérifie d'abord la reconnaissance de la clé publique. Si elle est inconnue, la transaction est immédiatement rejetée.

Même si le nœud malveillant connaît la clé publique, il doit réussir le protocole ZKP pour prouver la possession de la clé privée correspondante. L'échec à fournir une réponse correcte entraîne l'échec de la transaction et déclenche une erreur d'authentification. Ainsi, le système garantit que seuls les nœuds légitimes peuvent effectuer des transactions valides en utilisant des défis cryptographiques robustes et en rejetant les clés publiques inconnues.

Cet exemple montre comment l'application utilise l'authentification à l'aide de ZKP pour sécuriser les transactions dans un réseau blockchain. Il permet également aux utilisateurs de visualiser et d'interagir avec les données de manière sécurisée et transparente.

6. Défis et perspectives

L'utilisation du ZKP représente une avancée significative pour renforcer la sécurité et la confidentialité en permettant la vérification des informations sans les divulguer. Cependant, cette technologie rencontre des défis potentiels qui peuvent limiter son adoption et son efficacité. Néanmoins, les perspectives d'évolution des ZKP sont prometteuses et visent à surmonter ces obstacles, ouvrant ainsi la voie à une utilisation plus étendue. Cette section examine les principaux défis potentiels rencontrés par notre programme d'authentification dans la blockchain à l'aide de ZKP, tout en explorant les opportunités futures pour améliorer sa mise en œuvre et son intégration.

6.1. Défis Potentiels

- ✓ **Complexité du Protocole ZKP** : La mise en œuvre des preuves à divulgation nulle (ZKP) peut être complexe.
- ✓ **Scalabilité** : Le réseau peut devenir lent à mesure que le nombre de nœuds et de transactions augmente.
- ✓ **Consommation d'énergie** : Le minage basé sur le Proof of Work (PoW) est énergivore.
- ✓ **Validation des Transactions** : Actuellement, les transactions sont simplement ajoutées à une file d'attente et incluses dans un bloc sans vérification préalable.
- ✓ **Gestion des Clés Publiques** : La distribution et la gestion des clés publiques peuvent devenir complexes à mesure que le réseau grandit.

6.2. Prospectives et Améliorations

- ✓ **Complexité du Protocole ZKP** : Utiliser des bibliothèques ZKP éprouvées et effectuer des audits de sécurité réguliers.
- ✓ **Scalabilité** : Diviser le réseau en sous-réseaux plus petits et plus efficaces.
- ✓ **Consommation d'énergie** : Adopter des algorithmes de consensus plus écologiques comme PoS.
- ✓ **Validation des Transactions** : Implémenter un mécanisme pour vérifier la validité des transactions avant de les ajouter à un bloc, par exemple en s'assurant que l'expéditeur a suffisamment de fonds.
- ✓ **Gestion des Clés Publiques** : Utiliser des infrastructures à clé publique (PKI) pour une gestion plus efficace des clés.
- ✓ **Optimisation du Réseau** : Utiliser des techniques comme « le sharding » pour diviser le réseau en segments plus petits et gérer les transactions de manière plus efficace.
- ✓ **Analyse en Temps Réel** : Implémenter des outils d'analyse et de surveillance pour suivre les performances du réseau et détecter les anomalies en temps réel.

- ✓ **Tolérance aux Pannes** : Améliorer la résilience du réseau en ajoutant des mécanismes de redondance et de récupération après sinistre.

- ✓ **Interopérabilité avec d'Autres Technologies** : Intégrer des technologies comme l'Internet des Objets (IoT) ou l'intelligence artificielle (IA) pour étendre les cas d'utilisation de la blockchain.

- ✓ **Éducation et Formation** : Promouvoir l'éducation sur les concepts de blockchain et de ZKP.

- ✓ **Adoption de Standards** : Contribuer à l'élaboration de standards ouverts pour les protocoles de blockchain et de ZKP.

Ces défis et perspectives montrent les possibilités d'amélioration de notre programme, tout en surmontant les obstacles techniques actuels pour créer un réseau plus robuste et sécurisé.

7. Conclusion

Les preuves de zéro connaissance (ZKP) représentent une avancée majeure en cryptographie, offrant des protocoles qui préservent la confidentialité tout en garantissant l'intégrité des opérations. Leur utilisation dans des domaines tels que l'authentification, l'identification et l'échange de clés démontre leur polyvalence et leur pertinence.

Dans le cadre de notre recherche sur l'authentification dans la blockchain à l'aide de ZKP, ces protocoles ont joué un rôle central en assurant un niveau élevé de sécurité tout en préservant la confidentialité des utilisateurs. Leur implémentation a permis de relever des défis complexes tout en offrant des performances computationnelles optimisées, constituant ainsi une solution prometteuse pour les applications sécurisées dans le domaine de la blockchain.

Notre application d'authentification dans la blockchain à l'aide de ZKP renforce la sécurité des transactions et des identités dans les systèmes décentralisés. Nous avons créé une plateforme sécurisée et efficace pour authentifier les utilisateurs et garantir l'intégrité des données. Ce travail a abouti à une solution novatrice exploitant les avantages des ZKP, ouvrant la voie à de nouvelles opportunités dans le domaine de la sécurité informatique et de la protection des données personnelles.

En conclusion, les protocoles de zéro connaissance représentent une pierre angulaire dans le développement de systèmes cryptographiques avancés, ouvrant la voie à de nouvelles avancées dans la sécurisation des transactions et des communications en ligne. Notre solution d'authentification basée sur la blockchain et les ZKP constitue une avancée significative dans la réalisation de systèmes décentralisés sûrs et fiables.

CONCLUSION GENERALE

CONCLUSION GÉNÉRALE

Dans le domaine de la sécurité des réseaux informatiques, il est difficile de mettre en œuvre une solution qui répond parfaitement aux besoins de sécurité. Il est nécessaire de travailler continuellement pour développer des solutions permettant d'améliorer la sécurité des réseaux.

À travers notre étude consacrée à la double authentification de la blockchain à l'aide des Zero-Knowledge Proofs (ZKP), nous avons constaté les avantages significatifs de ces mécanismes pour le contrôle d'accès des utilisateurs aux services demandés, minimisant ainsi le risque des attaques menaçant le réseau. En combinant la blockchain avec la technologie des ZKP, nous avons pu créer une solution d'authentification qui assure un haut niveau de sécurité tout en préservant la confidentialité des utilisateurs.

L'implémentation de cette solution a démontré la capacité des ZKP à fournir une authentification forte sans révéler d'informations sensibles, ajoutant ainsi une couche supplémentaire de protection aux transactions et aux identités dans les systèmes décentralisés. En exploitant les principes de la blockchain et les protocoles cryptographiques avancés, nous avons développé une plateforme sécurisée et efficace pour authentifier les utilisateurs et garantir l'intégrité des données.

Notre projet s'est concentré sur la mise en place d'un mécanisme de double authentification utilisant des ZKP au sein de la blockchain. Ce travail a non seulement permis de relever des défis complexes liés à la sécurité des transactions numériques, mais a également ouvert la voie à de nouvelles perspectives dans le domaine de la protection des données personnelles et de l'authentification sécurisée. La solution mise en œuvre a offert des performances computationnelles optimisées, rendant les ZKP une option viable pour un large éventail d'applications dans la sécurité informatique.

En conclusion, ce mémoire a mis en évidence l'importance de l'authentification et des technologies de cryptage dans la sécurisation des systèmes informatiques, en particulier dans le contexte de la blockchain et des ZKP. Les recommandations fournies visent à guider l'intégration de ces technologies dans diverses applications, ouvrant ainsi la voie à des systèmes décentralisés plus sûrs et plus fiables

Les avancées technologiques réalisées grâce à ce projet, notamment l'utilisation combinée de la blockchain et des ZKP pour la double authentification, représentent une étape significative vers la réalisation de transactions numériques sécurisées et confidentielles. Elles offrent des solutions prometteuses pour répondre aux défis croissants de la sécurité informatique, ouvrant de nouvelles opportunités pour l'innovation dans ce domaine crucial. Nous espérons que ces travaux contribueront à améliorer la sécurité des réseaux informatiques et à promouvoir l'adoption de technologies de cryptage avancées pour protéger les données et les transactions en ligne.

Les perspectives d'amélioration de la blockchain via les preuves à divulgation nulle de connaissance (ZKP) sont prometteuses. Il est crucial d'utiliser des bibliothèques éprouvées et de réaliser des audits de sécurité réguliers. La scalabilité peut être améliorée par la division du réseau en sous-réseaux, tandis que la consommation d'énergie peut être réduite avec d’autres algorithmes de

consensus comme le PoS. De plus, l'interopérabilité avec d'autres technologies comme l'IoT et l'IA renforcera l'évolution généralisée de la blockchain.

Nous espérons que ce travail contribuera à améliorer la sécurité des systèmes distribués, notamment la Blockchain, et à promouvoir l'adoption des technologies avancées pour protéger les données et les informations en ligne.

ABSTRACT : In blockchain, users are identified by their public keys and use private keys to sign their transactions. Securing and managing private keys poses a crucial challenge for Blockchain applications, which are faced with the risk of user identity theft. Double authentication (2FA) offers a method of verifying the identity of users without disclosing their private keys, thereby strengthening the security of Blockchain transactions. However, its application in a transparent system such as the Blockchain requires innovative solutions to protect user confidentiality.

The Zero Knowledge Proof (ZKP) protocol is emerging as a promising solution to this challenge. Using ZKP, it is possible to prove that a user is authorised to access the Blockchain system without revealing sensitive information such as their private key. This thesis explores the joint application of 2FA and ZKP to secure user interactions with the Blockchain, particularly during unsecured connections, while guaranteeing data integrity and confidentiality.

Keywords : Blockchain, authentication, 2FA, ZKP, private key, public key, transaction, IT security.

ملخص: في مجال سلسلة الكتل (البلوكتشاين)، يتم تحديد هوية المستخدمين من خلال مفاتيحهم العامة واستخدام المفاتيح الخاصة لتوقيع معاملاتهم. ويشكل تأمين المفاتيح الخاصة وإدارتها تحديًا حاسمًا لتطبيقات البلوكتشاين التي تواجه خطر سرقة هوية المستخدم. وتوفر المصادقة المزدوجة (FA2) طريقة للتحقق من هوية المستخدمين دون الكشف عن مفاتيحهم الخاصة، وبالتالي تعزيز أمن المعاملات. ومع ذلك، يتطلب تطبيقها في نظام شفاف مثل البلوكتشاين حلولاً مبتكرة لحماية سرية المستخدم.

يبرز بروتوكول إثبات المعرفة المنعدمة كحل واعد لهذا التحدي. باستخدام بروتوكول (ZKP)، من الممكن إثبات أن المستخدم مخول بالوصول إلى نظام البلوكتشاين دون الكشف عن معلومات حساسة مثل مفتاحه الخاص. تستكشف هذه الأطروحة التطبيق المشترك للمصادقة المزدوجة (FA2) وبروتوكول (ZKP) لتأمين تفاعلات المستخدم مع البلوكتشاين، لا سيما أثناء الاتصالات غير الآمنة، مع ضمان سلامة البيانات وسريتها..

الكلمات المفتاحية: بلوكتشاين، إثبات المعرفة المنعدمة(ZKP) ، المصادقة المزدوجة(2FA) ، المفتاح الخاص، المفتاح العام، المعاملات، الأمن المعلوماتي

Résumé : Dans le domaine de la Blockchain, les utilisateurs sont identifiés par leurs clés publiques et utilisent des clés privées pour signer leurs transactions. La sécurisation et la gestion des clés privées posent un défi crucial pour les applications Blockchain, confrontées au risque d'usurpation d'identité des utilisateurs. La double authentification (2FA) offre une méthode pour vérifier l'identité des utilisateurs sans divulguer leurs clés privées, renforçant ainsi la sécurité des transactions Blockchain. Cependant, son application dans un système transparent comme la Blockchain requiert des solutions innovantes pour protéger la confidentialité des utilisateurs.

Le protocole de preuve à divulgation nulle de connaissance, Zero Knowledge Proof (ZKP), émerge comme une solution prometteuse à ce défi. En utilisant le ZKP, il est possible de prouver qu'un utilisateur est autorisé à accéder au système Blockchain sans révéler d'informations sensibles telles que sa clé privée. Ce mémoire explore l'application conjointe de la 2FA et du ZKP pour sécuriser les interactions des utilisateurs avec la Blockchain, notamment lors de connexions non sécurisées, tout en garantissant l'intégrité et la confidentialité des données.

Mots clés : Blockchain, authentification, 2FA, ZKP, clé privé, clé publique, transaction , sécurité informatique.

REFERENCES BIBLIOGRAPHIQUES

REFERENCES BIBLIOGRAPHIQUES

[1] Yaga, D., Mell, P., Roby, N., & Scarfone, K. (2018). "Blockchain Technology Overview." National Institute of Standards and Technology.

[2] Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System."

[3] Lesueur, M., Bironneau, L., Lux, G., & Morvan, T. (2020). "Réflexions sur les usages de la blockchain pour la logistique et le Supply Chain Management : une approche prospective." Avril 2022.

[4] Singhal, B., Dhameja, G., & Panda, P. S. (2018). "Beginning Blockchain: A Beginner's Guide to Building Blockchain Solutions." Apress.

[5] Xie, S., & Zheng, Z. (2017). "Blockchain Challenges and Opportunities: A Survey." International Journal of Web and Grid Services.

[6] Ganne, E. (2019). "Can Blockchain Revolutionize International Trade?" World Trade Organization.

[7] Puthal, D., Mohanty, S. P., Kougianos, E., & Yang, C. (2018). "The Blockchain as a Decentralized Security Framework." IEEE Consumer Electronics Magazine, 7(2), 18-21.

[8] Bitcoin.com. "Wiki: Block." Disponible sur : (https://bitcoin.com/wiki/Block).

[9] Gupta, A., & Gupta, S. (2018). "Blockchain Technology Application in Indian Banking Sector." Journal of Banking, Vol 19, Issue 2, July - December 2018, Pages: 75-84.

[10] Gatti, S., Houghton, T., & Rayport, J. (2004). "The Blockchain: The Future of Finance and Supply Chain Management." Harvard Business Review.

[12] Blockchain France. (2020). "Qu'est-ce que la Blockchain ?" Disponible sur : (https://www.blockchainfrance.net/). Consulté le : 20 janvier2024.

[13] Senat. (2018). "La Blockchain : Perspectives et Enjeux pour les Entreprises." Rapport d'information. Consulté le : 20 janvier2024.

[14] Borisov, N. (2011). "An Analysis of the Bitcoin Electronic Cash System." University of Illinois at Urbana-Champaign.

[15] Herbst, T. (2020). "Blockchain and GDPR: Contradictions and Solutions." European Journal of Law and Technology.

[16] Pass, R., Seeman, L., & Shelat, A. (2017). "Analysis of the Blockchain Protocol in Asynchronous Networks." Eurocrypt.

[17] Kiayias, A., Russell, A., David, B., & Oliynykov, R. (2017). "Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol." In Advances in Cryptology – eurocrypt 2017 (pp. 357-388). Springer.

[18] Larimer, D. (2014). "Delegated Proof-of-Stake (DPOS)." BitShares. Disponible sur : (https://bitshares.org/technology/delegated-proof-of-stake-consensus/). Consulté le :23 janvier2024.

[19] Sawtooth Lake Contributors. (2016). "Proof of Elapsed Time (PoET)." Sawtooth Lake Documentation. Disponible sur : (https://sawtooth.hyperledger.org). Consulté le : 23 janvier2024.

[20] Zhang, Y., Wen, Q., & Hu, J. (2018). "A Survey on Consensus Mechanisms and Mining Strategies for Blockchain." IEEE Access, 6, 69011-69026.

[21] Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). "Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction." Princeton University Press.

[22] Tapscott, D., & Tapscott, A. (2016). "Blockchain Revolution: How the Technology Behind Bitcoin is Changing Money, Business, and the World."

[23] Maull, R., Godsiff, P., Mulligan, C., Brown, A., & Kewell, B. (2017). "Distributed Ledger Technology: Applications and Implications." Strategic Change, 26(5), 481-489.

[24] Mukherjee, P. (2021). "Blockchain Technology: A Digital Revolution." Journal of Emerging Technologies, 10(1), 1-10.

[25] Buterin, V., & Wood, G. (2014). "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform." Ethereum White Paper. Disponible sur : (https://ethereum.org/whitepaper/). Consulté le : 26 janvier2024.

[26] Antonopoulos, A. M., & Wood, G. (2018). "Mastering Ethereum: Building Smart Contracts and DApps." O'Reilly Media.

[27] De Angelis, S., Aniello, L., Lombardi, F., & Margheri, A. (2018). "PBFT vs Proof-of-Authority: Applying the CAP Theorem to Permissioned Blockchain." Journal of Parallel and Distributed Computing, 119, 138-149.

[28] Macrinici, D., Cartofeanu, C., & Gao, S. (2018). "Smart Contract Applications within Blockchain Technology: A Systematic Mapping Study." Telematics and Informatics, 35(8), 2337-2354.

[29] Swan, M. (2015). "Blockchain: Blueprint for a New Economy." O'Reilly Media.

[30] Amazon Web Services. "Qu'est-ce que la Blockchain ?" Disponible sur : (https://aws.amazon.com/fr/what-is/blockchain). Consulté le : 24 janvier2024.

[31] Androulaki, E., et al. (2018). "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains." Proceedings of the Thirteenth EuroSys Conference.

[32] Brown, R. (2016). "Corda: An Introduction." R3 White Paper. Disponible sur : (https://www.corda.net/white-papers/). Consulté le : 02 fevrier 2024.

[33] Morgan, J. P., Chase, M., Iyer, P., Jensen, D., Lee, A., Potts, L., ... & Sharma, A. (2017). "Quorum: A Permissioned Implementation of Ethereum Supporting Data Privacy." J.P. Morgan White Paper. Disponible sur : (https://www.goquorum.com/). Consulté le : 02 fevrier 2024.

[34] MEF : Economie.gouv.fr. Système Centralisé Vs Système Décentralisé. Disponible sur : (https://www.economie.gouv.fr/). Consulté le : 22 janvier 2024.

[35] Decker, C., & Wattenhofer, R. (2013). "Information Propagation in the Bitcoin Network." IEEE P2P 2013 Proceedings.

[36] Whitman, M. E., & Mattord, H. J. (2013). Principles of Information Security. Cengage Learning.

[37] Tipton, H. F., & Krause, M. (2013). Information Security Management Handbook, Sixth Edition, Volume 1. CRC Press.

[38] Jiang, N., Ren, K., & Sun, X. (2020). A Secure Authentication Scheme Based on Physical Unclonable Function for Industrial Internet of Things. IEEE Transactions on Industrial Informatics, 16(2), 1316-1324.

[39] Dilu Nlemvo, S. (2014). Implantation d'un système de gestion des utilisateurs et monitoring des taches dans un réseau d'entreprise, Université de Kinshasa.

[40] Phillips, T. E., & Housley, R. (2005). Smart Card Handbook: Third Edition. Wiley.

[41] Maltoni, D., Maio, D., Jain, A. K., & Prabhakar, S. (2009). Handbook of Fingerprint Recognition. Springer Science & Business Media.

[42] Diedhiou, M. (2014). Authentification du réseau basé sur les certificats, Université Assane Seck de Ziguinchor.

[43] Mudikolele, J. T. (2016). Mise en oeuvre d'un système distribué pour l'identification et le suivi du casier judiciaire, Université pédagogique nationale.

[44] Assih, E. (2019). Déploiement de la solution de sécurité Firewall, Authentification et Autorisation RADIUS sur un réseau local d'entreprise. Mémoire de Master, Université de Lomé, Togo.

[45] Kent, S., & Seo, K. (2008). Security Architecture for the Internet Protocol. RFC 4301.

[46] NIST. (2017). Digital Identity Guidelines: Authentication and Lifecycle Management. Special Publication 800-63B.

[47] Forouzan, B. A. (2013). Data Communications and Networking. McGraw-Hill Education.

[48] Tanenbaum, A. S., & Van Steen, M. (2006). Distributed Systems: Principles and Paradigms. Pearson Education.

[49] Sasse, A., Furnell, S., & Harbach, M. (2020). Authentication Technologies and Their Implications for Privacy.

[50] Raissi, A. (2010). Authentification dans les Réseaux Wifi par le protocole radius, édition 2010.

[51] Tighilt D, & Hamoudi A. (2012). Mise en place d'une solution d'authentification RADIUS. Univ Abderrahmane Mira de Bejaia.

[52] Simpson, W. (2014). PPP Challenge Handshake Authentication Protocol (CHAP). RFC 1994.

[53] Microsoft Corporation. (2003). Microsoft Challenge Handshake Authentication Protocol (MS-CHAP). Microsoft Developer Network (MSDN). Consulté le : 31 janvier 2024.

[54] Certa, C. (2012). La sécurité des systèmes d'information. édition 2012.

[55] Microsoft Corporation. (2005). MS-CHAP-v2: Microsoft Challenge Handshake Authentication Protocol Version 2. Microsoft Developer Network (MSDN). Consulté le : 31 janvier 2024.

[56] Singh, P. (2014). Analysis of Security Issues and Their Solutions In Wireless LAN!. ICICES2014 - S.A. Engineering College, Chennai, Tamil Nadu, India.

[57] Zidani, F. (2018). Solution d'authentification et de gestion de clés pour le standard 802.11i des réseaux WiFi, Université Ferhat Abbas, Setif.

[58] Chen, J., Jiang, M., & Liu, Y. (2005). Wireless LAN Security and IEEE 802.11i. National Tsing Hua University, IEEE Wireless Communications.

[59] Saccavini, L. (2003). 802.1X et la sécurisation de l'accès au réseau local. Direction des Réseaux et Systèmes d'Information, INRIA.

[60] Boukoucha, H., & Bouktit, N. (2021). La mise en œuvre d'un système d'authentification unique SSO avec CAS. Mémoire de Master Université Abderrahmane Mira, Bejaia.

[61] Kherbache, M., & Letat, Z. (2015). Proposition et implémentation d'un protocole d'authentification unique. Mémoire de fin de cycle, université de Bejaia.

[62] Maudoux, C. (2018). Implémentation de l'authentification à double facteur dans la solution de SSO AAA LemonLDAP:NG 2.0.

[63] Mouté Nyokon, C. E. (2011). Authentification et protocole PPPOE: le cas de l'accessibilité à l'internet via ringodialeré, Université de Yaoundé I.

[64] Gauthier, J. (2018). Remote Authentication Dial In User Service. Université Gustave Eiffel.

[65] Bordères, S. (2006). Authentification réseau avec radius 802.1x - eap - freeradius- Collection Blanche. Eyrolles Edition.

[66] Rigney, C., Willens, S., Rubens, A., & Simpson, W. (2000). Remote Authentication Dial-In User Service (RADIUS). RFC 2865.

[67] Aboba, B., & Zorn, G. (2004). Extensible Authentication Protocol (EAP) Overview. RFC 3748.

[68] Stallings, W. (2016). Network Security Essentials: Applications and Standards. 4th edition, Pearson Education.

[69] Schnorr, Claus-Peter. (1989). Efficient Identification and Signatures for Smart Cards. Advances in Cryptology — CRYPTO '89 Proceedings, Springer.

[70] Sain, M., Patel, P., et al. (2020). Public Key Infrastructure (PKI) in Blockchain: Applications, Challenges, and Solutions. IEEE Access.

[71] Parecki, A. (2012). OAuth 2.0 Simplified. [En ligne] Disponible sur : oauth.net. Consulté le : 12 mars 2024.

[72] Bertocci, V. (2017). OpenID Connect explained. [En ligne] Disponible sur : auth0.com. Consulté le : 12 mars 2024.

[73] Auth0. (2018). JSON Web Token (JWT). [En ligne] Disponible sur : jwt.io. Consulté le : 12 mars 2024.

[74] W3C. (2020). Decentralized Identifiers (DIDs). [En ligne] Disponible sur : w3.org. Consulté le : 12 mars 2024.

[75] Allen, C. (2016). Self-Sovereign Identity: A Brief Introduction. [En ligne] Disponible sur : www.lifewithalacrity.com. Consulté le : 28 mars 2024.

[76] Green, M. (2018). Zero-Knowledge Proofs: An illustrated primer.Cryptography Engineering Blog, 27 nov. [En ligne] Disponible sur : blog.cryptographyengineering.com. Consulté le : 2 avril 2024.

[77] Goldwasser, S., Micali, S., & Rackoff, C. (1989). The Knowledge Complexity of Interactive Proof Systems. SIAM Journal on Computing, 18, 186-208. (Preliminary version in 17th STOC, 1985).

[78] Blum, M., Feldman, P., & Micali, S. (1988). Non-Interactive Zero-Knowledge and Its Applications. Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC '88), 103-112. DOI: 10.1145/62212.62222.

[79] Goldreich, O., Micali, S., & Wigderson, A. (1991). Proofs that Yield Nothing But their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. Journal of the ACM, 38(3), 691-729. DOI: 10.1145/116825.116852.

[80] Hasan, J. (2019). Overview and Applications of Zero Knowledge Proof (ZKP).

[81] Tom. (2022). What is Zero-knowledge proofs? Coinmonks. Disponible sur : https://medium.com/coinmonks/what-is-zero-knowledge-proofs-9c1c62e4ecdc. Consulté le : 25 mars 2024.

[82] Niranjanamurthy, M., Shashank, K. S., Sumanth, P. Gowda, & Suhas Bhatta, S. (2016). Research Study on Two Factor Zero Knowledge Proof Authentication System. Thèse de doctorat, MSRIT, Bangalore, Inde.

[83] Nguyen, Quan, Rudoy, Mikhail, Srinivasan, Arjun. (2014). Two Factor Zero Knowledge Proof Authentication System. 6.857 Spring 2014 Project.

[84] Hallot, E. (2024). Zero Knowledge Proof: fonctionnement et cas d'usages. Académie, Technologie.dispo sur : https://www.crypto-patrimoine.fr/academie/zero-knowledge-proof/ . Consulté le : 29 mars 2024.

[85] Dwivedi, A. D., Singh, R., & Ghosh, U. (2021). Privacy preserving authentication system based on non-interactive zero knowledge proof suitable for Internet of Things. Thèse de doctorat, Aalborg University, Copenhagen Business School.

[86] Gradwohl, R., Naor, M., Pinkas, B., & Rothblum, G. N. (2009). Cryptographic and Physical Zero-Knowledge Proof Systems for Solutions of Sudoku Puzzles. Theory of Computing Systems, 44(2), 245-268.

[87] Jo, T. (2019). An Exploration of Zero-Knowledge Proofs and zk-SNARKs. EAS499. Advisor: Professor David Crosbie.

[88] Groth, J., & Sahai, A. (2008). Efficient Non-interactive Proof Systems for Bilinear Groups. Eurocrypt.

[89] Momin, Krinza. (2023, April 11). A Non-Mathematical Introduction to Zero Knowledge Proof. Disponible sur : https://kayprasla.medium.com/a-non-mathematical-introduction-to-zero-knowledge-proof-c1a4a269e308 . Consulté le : 04 avril 2024.

[90] Bouchakour, E H. (2024).Cours 06 pour master CSIA: Zero Knowledge Proof", Section : Protocole de Schnorr, Université Ain Témouchent, page 23. Disponible sur :https://classroom.google.com/c/NjI2OTQ2ODkzMzYz?pli=1 .consulté le 06 février 2024

[91] Feige, U., Fiat, A., & Shamir, A. (1986). Zero-Knowledge Proofs of Identity. Advances in Cryptology – CRYPTO' 86. doi:10.1007/3-540-47721-7_14

[92] Cao, Z., Wang, X., & Liu, L. (2020). A Note on One Popular Non-Interactive Zero-Knowledge Proof System. International Journal of Network Security, 22(4), 681-685.

[93] Noël, Aymeric. (2021). Étude de cas d'utilisation de Zero Knowledge Proof (ZKP). Disponible sur : https://blog.octo.com/etude-de-cas-dutilisation-de-zero-knowledge-proof-zkp. Consulté le : 08 avril 2024.

[94] ZK-SNARKs vs ZK-STARKs vs Bulletproofs. Ethereum Stack Exchange, stackexchange.com/questions/59145/zk-snarks-vs-zk-starks-vs-bulletproofs-updated/63778#63778.

Consulté le : 05 avril 2024.

[95] Bonneau, J., Boneh, D., Gennaro, R., & Goldfeder, S. (2015). On Bitcoin as a public randomness source. Cryptology ePrint Archive, Report 2015/1015.

[96] Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. Proceedings of the IEEE Symposium on Security and Privacy.

[97] Wang, H., Xu, R., Su, H., & Jia, C. (2019). Towards blockchain-based audit and compliance management systems. Future Generation Computer Systems, 100, 114-122.

[98] Larimer, D. (2014). Delegated Proof-of-Stake (DPOS). BitShares. Disponible sur : https://bitshares.org/technology/delegated-proof-of-stake-consensus/. Consulté le : 28 mars 2024.

[99] Doe, J., & Smith, J. (2020). Zero-Knowledge Proofs: A Comprehensive Overview and Technical Analysis. Journal of Cryptography.

[100] Chandra, Amit. (2023). Zero-Knowledge Proof: Enhancing Privacy and Security in the Digital Age. LinkedIn, https://www.linkedin.com/pulse/zero-knowledge-proof-enhancing-privacy-security-digital-amit-chandra/ Consulté le : 04 avril 2024.