MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABDELHAMID IBN BADIS - MOSTAGANEM

Faculté des Sciences Exactes et d'Informatique

Département de Mathématiques et informatique

Filière : Informatique

MEMOIRE DE FIN D'ETUDES

Option : **Ingénierie des Systèmes d'Information**

THEME : Les questions de confidentialité dans les organisations décentralisées sur la blockchain

Encadré par : **Dr MIROUD Mohammed El Mustapha**

Réalisé par :

KHELLAF Mohamed El Amine

BENHEBBA Amin

Membre de jury :

Président :
Dr HABIB ZAHMANI .M
Examinateur :
Dr BENSAOUD Karim

Année Universitaire 2022-2023

Dédicaces

Je dédie ce modeste travail à :

Mes chers parents pour leur soutien indéfectible et tous les sacrifices qu'ils ont consentis pour que je réussisse. Votre amour inconditionnel et vos encouragements constants ont été ma source de motivation.

Je le dédie également à :

Ma femme. Mes frères et sœurs ainsi que leurs enfants. Je dédie ce travail à ma nièce Sirine et à ma nièce Rim, que j'aime énormément. Votre joie de vivre et votre présence précieuse ont enrichi ma vie d'une manière unique.

Que cette dédicace témoigne de ma profonde reconnaissance et de mon amour envers vous tous. Votre soutien a été essentiel et je suis honoré(e) de vous avoir dans ma vie.

Mohamed el Amine KHELLAF

Dédicaces

Je dédie ce modeste travail à :

Mes chers parents pour tous les sacrifices qu'ils ont consentis pour que je réussisse.

Je le dédie également à : Ma femme. Mes frères et sœurs ainsi que leurs enfants et toute ma famille, mes amis et collègues.

En un mot à tous les gens qui ont contribué à ma réussite de près ou de loin.

Je remercie également tous mes professeurs pour leurs efforts.

Amine BENHEBBA

Remerciements

Tout d'abord, on tient à remercier le bon dieu de la force et du courage qu'il nous a accordés pour mener à bien ce modeste travail, ainsi que nos petites familles pour l'inconditionnel soutien qu'il nous généreusement offert.

On tient à remercier tous ceux et celles qui ont contribué à finaliser ce modeste travail.

Nos remerciements vont aussi à monsieur MIROUD Mohamed el Mustapha, notre encadreur pour nous avoir guidé pour la réalisation de ce projet.

On tient également à remercier tout l'équipage de la faculté des sciences exactes et de l'informatique FSEI Mostaganem, tous les enseignants et toutes les enseignantes sans oublier l'ensemble du personnel.

Enfin, on tient à remercier les étudiants Master 2 Informatique ISI pour leur aide morale et leur esprit positif pendant toute la période scolaire.

Mohamed et Amine KHELLAF, Amine BENHEBBA

Résumé

L’objectif principal de notre projet de fin d'étude est d'approfondir notre compréhension des organisations décentralisées et de leur confidentialité. Nous avons commencé par explorer la technologie blockchain, qui est une technologie décentralisée, et nous avons étudié son fonctionnement. Ensuite, nous avons discuté des aspects des organisations décentralisées autonomes, tels que leurs caractéristiques et leur intérêt, ainsi que des plateformes qui peuvent garantir la confidentialité des données. Bien que les DAO offrent une prise de décision transparente, la divulgation d'informations sensibles peut compromettre la vie privée des membres et des activités de l’organisation. Cependant, des solutions telles que les preuves à divulgation nulle de connaissance, comme le ZK-SNARK, peuvent préserver la confidentialité tout en garantissant la vérifiabilité. Dans ce rapport, nous avons présenté un protocole visant à résoudre les problèmes de transparence financière des DAO tout en offrant des fonctionnalités de confidentialité.

Mots-clés :

Blockchain, Bitcoin, cryptographie, cryptomonnaies, consensus, Ethereum, Smart contracts, Organisations Décentralisées Autonomes (DAO).

Abstract

The main objective of our end-of-study project is to deepen our understanding of decentralized organizations and their privacy. We started by exploring blockchain technology, which is a decentralized technology, and studied its functioning. Then, we discussed aspects of decentralized autonomous organizations (DAOs), such as their characteristics and benefits, as well as platforms that can ensure data privacy. While DAOs offer transparent decision-making, the disclosure of sensitive information can compromise the privacy of members and organizational activities. However, solutions such as zero-knowledge proof systems, like ZK-SNARK, can preserve privacy while ensuring verifiability. In this report, we presented a protocol aimed at addressing the financial transparency issues of DAOs while offering privacy features.

Keywords:

Blockchain, Bitcoin, cryptography, cryptocurrencies, consensus, Ethereum, Smart contracts, Decentralized Autonomous Organizations (DAO).

Liste des figures

Liste des tableaux

Table des matières

Introduction Générale

Les blockchains, un terme de plus en plus courant de nos jours; l'inception de l'idée était premièrement dévoilée par un inconnu *Satoshi Nakamoto* qui a révolutionné la façon dont on pense au gouvernance, gestion, et construction de nos systèmes d'information, systèmes financiers, et même la manière dont on échange les biens et les fonds monétaires d'une façon tout à fait décentralisée. Cette nouvelle technologie a rapidement séduit un nombre très important d'enthousiaste et des communautés qui s'y trouvaient une ouverture assez prometteuse challengeant les anciens systèmes basés sur une autorité centrale gouvernante et non questionnable. S'appuyant sur les techniques cryptographiques déjà bien élaborées pour assurer la sécurité sur les échanges internet, cette nouvelle idée s'est montrée dès sa création comme un nouveau client de ces techniques cryptographiques mais contrairement à ces rôles préalables, elles ont été principalement utilisées pour assurer l'intégrité des données hébergées dans les réseaux blockchains. Cette technologie a été vastement sollicitée pour répondre à différentes exigences des domaines s'appuyant sur les architectures peer-to-peer où aucune partie ne peut être confiée la gouvernance et la gestion du réseau , entre autres les cryptomonnais, les Défi, les jeux vidéo,...etc. dernièrement, on assiste à l'apparition d'un nouveau secteur profitant des même idée de décentralisation sous le nom d'organisation autonome décentralisé (DAO, autrement connu sous le jargon anglais *Decentralized Autonomous Organisations*). Les organisations décentralisées autonomes sont devenues un sujet de plus en plus populaire ces dernières années en raison de leur capacité à démocratiser les décisions d'affaires et à réduire les barrières à l'entrée pour les participants. Leur potentiel pour éliminer la nécessité d'intermédiaires et d'assurer une transparence accrue a attiré l'attention de nombreux acteurs du monde des affaires et de la technologie.

Ce rapport de projet se concentre sur l'étude des DAO, ses mécanismes, ainsi que ses domaines d'application et leurs modes de fonctionnement de façon plus ou moins détaillée, il représente une réponse aux questions de confidentialité souvent lever a cause de la nature de la technologie qu'elles se base dessus.

Lors du premier chapitre, nous explorerons en détail les DAO (Organisations autonomes décentralisées) en nous concentrant sur l'exemple de la Blockchain Ethereum. Le premier chapitre se penchera sur la structure et le fonctionnement des DAO, ainsi que sur les principes fondamentaux de la technologie de la Blockchain. Nous aborderons également les mécanismes de gouvernance qui permettent aux DAO de fonctionner de manière autonome. En outre, nous examinerons les défis clés auxquels sont confrontés les DAO et examinerons des exemples d'entités qui peuvent être considérées comme des DAO.

Dans le deuxième chapitre, nous approfondirons notre compréhension des DAO en examinant leur intérêt, leur fonctionnement et les problèmes de confidentialité associés. Nous explorerons diverses techniques de confidentialité dans les blockchains, telles que le "commitment" pour masquer certaines informations et les preuves à divulgation nulle de connaissance, avec un accent particulier sur le ZK-SNARK. Cette technique puissante permet de prouver la connaissance d'une information sans la divulguer.

Dans le troisième chapitre, nous nous concentrerons sur l'exemple concret de la ConstitutionDAO pour examiner les problèmes de confidentialité dans les DAO. Nous étudierons les objectifs de cette DAO et analyserons en détail le cycle de vie de sa trésorerie gérée sur la plateforme Ethereum. Ce cycle comprendra trois étapes principales : la création du DAO, le dépôt de fonds et le retrait de ces fonds.Nous fournirons une description détaillée de chacune de ces étapes, mettant en évidence les mesures mises en place pour garantir la confidentialité des informations, protéger les fonds et permettre une gestion privée des actifs par le gestionnaire du DAO.

Ce document se veut être une humble étude, pour les futures étudiant de la faculté en matière des organisations autonomes décentralisées, où nous avons essayé d'exposer un champ d’application de la technologie blockchain qui s’avère être assez prometteuse, et de donner une brève agrégation des problèmes que cette technologie rencontre en ce qui concerne la préservation de la confidentialité.

Chapitre 1

Technologie Blockchain

1.1 Introduction

Ce chapitre explorera en détail la structure et le fonctionnement des DAO en se concentrant sur l'exemple de la Blockchain Ethereum[2]. Nous allons couvrir les principes fondamentaux de la technologie de la Blockchain, ainsi que les mécanismes de gouvernance qui permettent aux DAO de fonctionner de manière autonome. Nous allons également aborder les défis clés auxquels les DAO sont confrontés, en examinant des exemples d'entités qui peuvent être définies comme DAO

1.2 La technologie blockchain

1.2.1 Caractéristiques générales

Lorsqu'on mentionne la technologie blockchain[1], le bitcoin[3] est généralement la première chose qui vient à l'esprit. La raison est que la crypto-monnaie bitcoin était la première application de cette nouvelle technologie lorsqu'elle a été lancée en 2009. Cependant, la technologie blockchain, qui est une forme de technologie de registre distribué (DLT), a maintenant trouvé de nombreuses autres applications que le bitcoin en offrant un nouveau moyen de partager en toute sécurité les données à travers un système de consensus distribué sans confiance fonctionnant sur un réseau de nœuds qui peut être utilisé dans de nombreux contextes différents. Dans sa forme la plus basique, la technologie blockchain peut être décrite comme "une base de données chronologique des transactions enregistrées par un réseau d'ordinateurs". Le terme "blockchain" fait référence à ces transactions qui sont placées dans des blocs qui sont liés les uns aux autres, formant une blockchain. Les caractéristiques de la blockchain ont hissé le DLT à un tout nouveau niveau

et les nombreuses nouvelles possibilités qu'elle offre dans l'espace numérique sont véritablement révolutionnaires[4].

1.2.2 Structure de la blockchain

La blockchain est un système de stockage qui ne permet que l'ajout de nouvelles informations et forme une séquence chronologique. Il n'est pas possible de modifier ou supprimer des informations existantes, mais seulement de les ajouter. L'utilisation de fonctions cryptographiques garantit l'intégrité des informations et la préservation de l'historique des changements, garantissant ainsi la vérifiabilité et la transparence du système[5].

Les informations stockées sur la blockchain peuvent être diverses et sont souvent décrites comme des transactions, telles que des transferts de cryptomonnaie, des reconnaissances officielles, des accords entre personnes. Les transactions sont regroupées en blocs semblables aux pages d'un registre papier. Chaque bloc comprend un entête avec des informations de gestion et une liste de transactions, ainsi que le hash du bloc précédent[5].

La **Figure 1** montre le mécanisme de la blockchain. Le premier bloc(n) contient deux transactions et le hash du bloc précédent (n-1). Le hash du bloc(n) est : 022656E2. Le bloc suivant (n+1) contient également deux transactions et inclut le hash du bloc précédent (n).

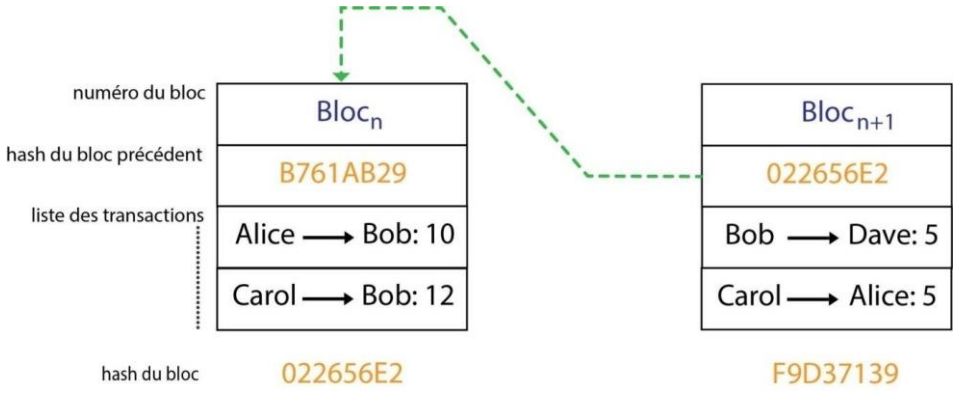

Figure 1 : Chaînage Des Blocs A L'aide Du Hash[5]

Comme montré sur la **Figure 2** au-dessous, si l'un des nœuds altère une transaction dans le bloc n, ce qui modifie par exemple le montant de la seconde transaction, cela entraîne un nouveau hash (83ecf615) qui ne correspond plus à celui stocké dans le bloc suivant (022656E2), ce qui permet la détection de la fraude. Pour rétablir la cohérence de la chaîne, il serait nécessaire de recalculer tous les blocs suivants, ce qui est rendu pratiquement impossible grâce au fonctionnement de la blockchain.

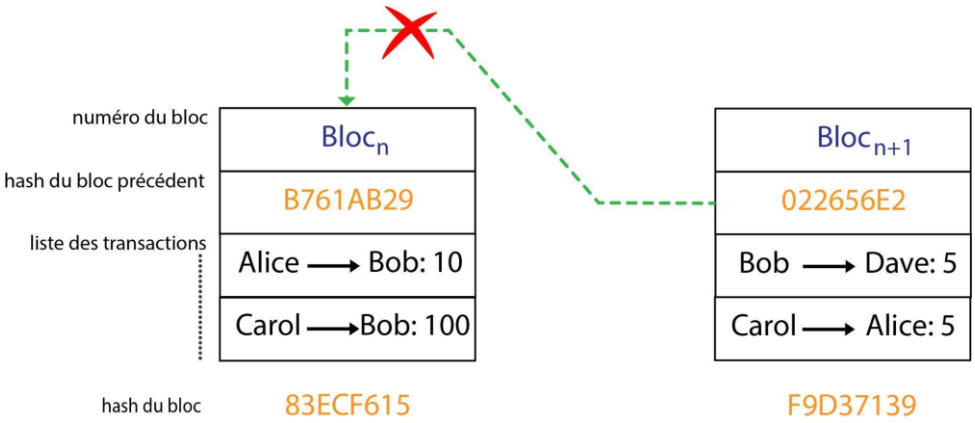

Figure 2 : Incohérence des liens entre blocs suite à une modification d'une transaction[5].

Ce mécanisme de chaînage des blocs garantit donc l'intégrité et l'immuabilité des données stockées sur le bloc.

1.2.3 Types de blockchain

Le concept original de la blockchain impliquait une validation décentralisée et publique par l'ensemble des nœuds du réseau. Cependant, certaines institutions ont exprimé des préoccupations quant à la nature publique de la blockchain. En réponse à ce besoin, de nouveaux concepts ont émergé, tels que les blockchains privées et de consortium. Une blockchain de consortium restreint la validation des blocs à un nombre défini de nœuds, avec la possibilité de limiter l'accès à certains nœuds pour la lecture de l'ensemble de la blockchain, rendant la consultation privée. Alternativement, la consultation peut être publique pour tous les nœuds du réseau. Une blockchain peut également être partitionnée, avec certaines données consultables publiquement et d'autres disponibles uniquement sur un réseau privé, offrant ainsi une consultation hybride. En revanche, une blockchain privée

réserve le processus de validation à un seul acteur, et les consultations peuvent être privées, publiques ou hybrides.[6].

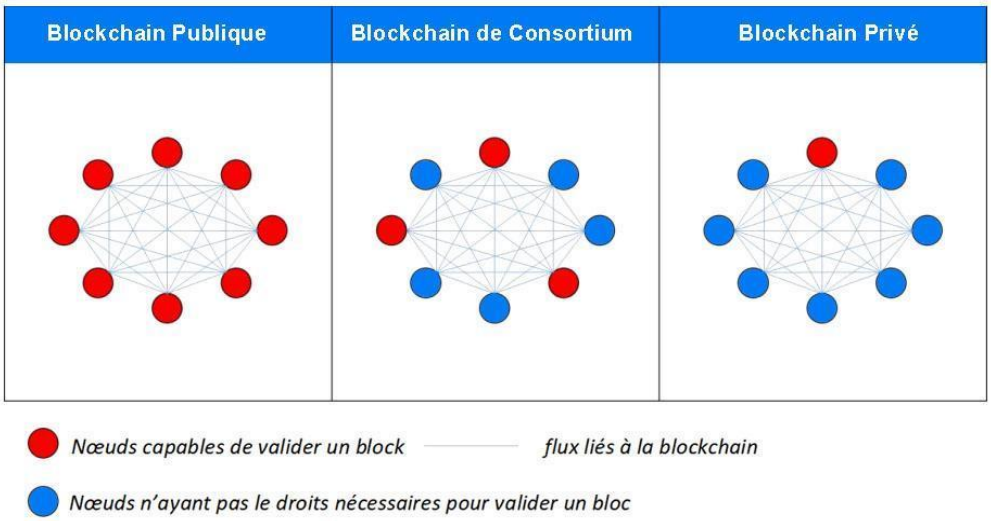

Figure 3 : Types de Blockchain selon le mode de validation de blocs[6].

Chaque point dans ce schéma représente un nœud du réseau. On peut y voir la différence de gouvernance entre les différents types de blockchain (publique, privée, ou de consortium).

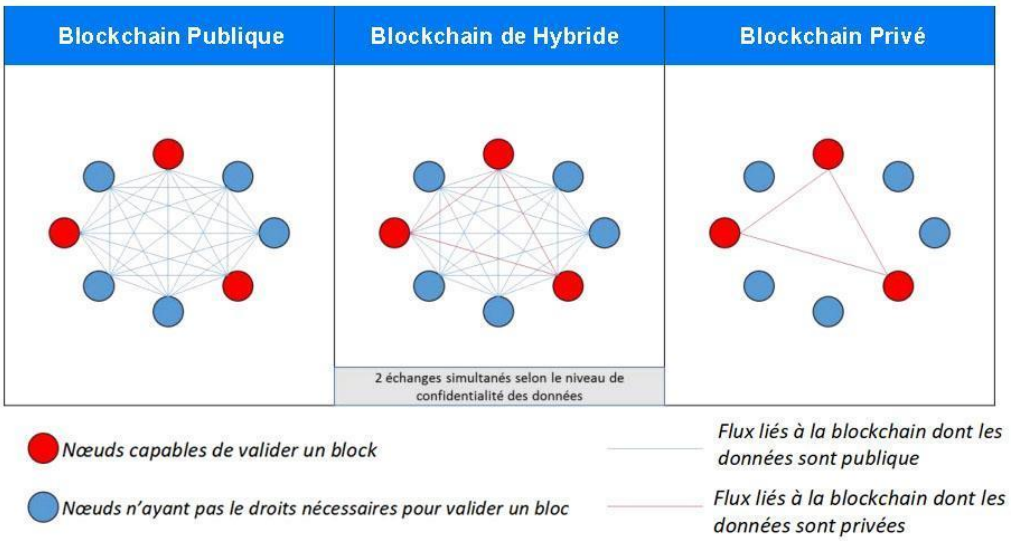

Figure 4 : Partage des données d'une Blockchain de Consortium sur le réseau[6].

1.3 La blockchain comme mécanisme d’exécution

L'évolution de la blockchain a conduit à des services d'exécution de code garantissant l'exécution automatique et inviolable des programmes dans des conditions spécifiques, appelés "smart contracts".

1.3.1 Smart contracts

Le concept de smart contrat a été proposé par Nick Szabo, qui a postulé que les clauses contractuelles peuvent être codées dans le logiciel pour éviter tout manquement au contrat. Selon Szabo, un smart contract doit être conçu pour satisfaire les quatre critères suivants :

- La possibilité pour les parties d’observer la manière dont les autres exécutent leur part du contrat, et de leur prouver la bonne réalisation de leurs propres obligations.
- Des éléments probants concernant la bonne exécution du contrat ou au contraire relatif à une brèche dans celui-ci doivent être disponibles pour un arbitre tiers.
- L’information concernant le contrat lui-même et son exécution doit être accessible uniquement aux parties qui y ont un intérêt légitime.
- Le caractère exécutoire du contrat doit être garanti[5].

1.4 Organisations autonomes décentralisées

Les organisations autonomes décentralisées (DAO) sont une nouvelle forme d'organisation qui utilise la technologie de la blockchain pour automatiser certaines fonctions clés telles que la prise de décision, la formation de capital et le déploiement de capital. Elles offrent de nombreux avantages potentiels, tels que la transparence, l'immuabilité, la démocratie décentralisée et l'efficacité accrue. Les DAO sont considérées comme une solution prometteuse pour les organisations de l'âge numérique et pourraient révolutionner la manière dont les entreprises sont gérées à l'avenir[4].

L'idée d'une entité décentralisée et autonome exécutée sur une blockchain provient du billet de blog de l'entrepreneur de la blockchain Daniel Larimer sur les coûts cachés de

Bitcoin, publié le 7 septembre 2013. Daniel Larimer faisait valoir qu'une crypto-monnaie pouvait être considérée comme une Société Autonome Décentralisée (DAC), où le code source représente les statuts et les détenteurs de jetons sont des actionnaires. Dans sa comparaison, Daniel Larimer a poursuivi en disant que le but de la DAC est de maximiser la valeur pour ses détenteurs de jetons en effectuant des activités[4].

1.4.1 Concept de DAO

Le concept de DAO (Organisation décentralisée autonome) est apparu en 2013 avec la publication du livre blanc de Vitalik Buterin, co-fondateur d'Ethereum. Les DAOs sont des organisations décentralisées qui utilisent la technologie de la blockchain pour permettre aux utilisateurs de prendre des décisions de manière démocratique et transparente, sans avoir besoin d'un tiers centralisé. Les contrats intelligents sont utilisés pour automatiser les processus de décision et d'exécution, ce qui rend les DAOs plus rapides, plus sécurisées et plus fiables que les organisations traditionnelles[4].

1.4.2 Définition d'une DAO

En général, la définition d'une DAO n'est pas homogène ni parmi la communauté informatique ni parmi la communauté juridique. Cependant, certaines caractéristiques récurrentes peuvent indiquer ce qui est généralement compris comme une DAO. La caractéristique la plus importante est que la DAO est une forme d'entité organisée, comparée à certaines entreprises. Chaque définition mentionne que les DAO accomplissent des tâches en fonction de certaines règles de gouvernance similaires à des statuts. Cela indique qu'il est compris que les DAO sont organisés à l'intérieur et capables d'accomplir des tâches ayant des impacts externes. Du côté technique, certains relient directement les DAO aux contrats intelligents et à la technologie blockchain, indiquant implicitement que les DAO sont un réseau de contrats intelligents fonctionnant sur une blockchain. Certains auteurs affirment même que les DAO ne peuvent fonctionner que sur une blockchain publique et ouverte. D'autres essaient de souligner les aspects techniques plus généraux des DAO en les décrivant comme un logiciel fonctionnant sur un réseau peer to peer cryptographiquement sécurisé[13].

Nous pouvons trouver la définition plus générale suivante dans la référence [13] :

Un DAO est définie comme l'entité créée par le déploiement d'un logiciel autonome sur un système distribué, permettant à un réseau de participants d'interagir et de gérer les ressources de manière transparente selon les règles définies par le code du logiciel.

1.5 Aspects techniques

Les DAO sont des organisations qui fonctionnent sur la base de contrats intelligents et de la blockchain. Les aspects techniques clés des DAO incluent :

1.5.1 Processus de prise de décision dans les DAO

Les organisations autonomes décentralisées (DAOs) sont des entités décentralisées telles que des corporations et des institutions qui fonctionnent entièrement de manière autonome et décentralisée sur une blockchain. Les décisions importantes de gestion sont prises par la logique définie dans le code, exécutée par des contrats intelligents et, finalement, grâce à l'automatisation de la prise de décision, créant de la valeur pour les clients De plus, dans les DAOs, il peut y avoir des tâches répétitives et simples qui peuvent être automatisées et des tâches qui nécessitent une réflexion créative, une innovation et une responsabilité[14].

Ce type de tâches nécessite également des engagements humains, qui sont également incités par le code.

Les organisations doivent remplir les exigences suivantes pour se définir comme une DAO :

- **Tokens de transaction** : une DAO doit contenir une propriété précieuse sous forme de tokens, qui peuvent être utilisés pour les récompenses et qui peuvent représenter le pouvoir de vote de chaque membre.
- **Exécution autonome** : la DAO agit de manière autonome et ne peut être influencée par des forces externes. L'organisation et son code source ouvert sont entièrement

transparents et donc incorruptibles. Les fonctionnalités et les règles du programme sont écrites dans le code et maintenues sur la blockchain.

- **Consensus** : les parties prenantes doivent être totalement d'accord si des décisions doivent être prises, par exemple si un membre veut retirer des fonds. De plus, les bugs et les optimisations nécessitent une décision et un vote démocratique, qui peuvent être considérés comme des problèmes de sécurité s'ils sont corrigés trop tard.
- **Contractants** : afin d'accomplir les tâches et d'atteindre les objectifs commerciaux, la DAO embauche des fabricants, des développeurs et d'autres contractants en fonction des décisions des actionnaires.
- **Propositions** : le processus de prise de décision commence toujours avec une proposition. Une proposition peut également nécessiter un dépôt monétaire pour éviter une surcharge du réseau de propositions[14].
- **Vote** : cette étape est cruciale, car des activités telles que la levée de fonds, l'investissement et la réparation ne peuvent avoir lieu que si une majorité est atteinte.

Le processus de vote dans les DAO offre également une alternative à la situation actuelle de la prise de décision hiérarchique et de gestion dans les entreprises traditionnelles. Une DAO peut avoir différentes parties prenantes et chacun aura le droit de vote sur la façon d'allouer les fonds de l'organisation. Les fonds peuvent être utilisés pour payer des salaires ou comme devise interne pour récompenser le travail. Les droits de vote peuvent être distribués soit en fonction de la quantité de jetons possédée ou de manière équitable[14].

1.5.2 Les contrats intelligents

Du point de vue technique, une DAO représente un ou plusieurs contrats intelligents, qui sont exécutés sur la technologie de consensus distribué telle que la blockchain Ethereum. Ethereum offre une blockchain avec un langage de programmation turing-complet intégré, où chacun peut construire des applications décentralisées basées sur ses propres règles de propriété, formats de transaction et fonctions de transition d'état,

qui s'exécutent lorsque certaines conditions sont remplies. Les frais de transaction pour les contrats intelligents sont payés en Ether[15], le jeton d'Ethereum[14].

Les messages sur Ethereum peuvent être des transactions exécutées par des entités externes ou des contrats internes et peuvent contenir des données pour le destinataire. Un message envoyé de manière externe est une transaction, contenant le destinataire du message, la signature de l'expéditeur, le montant envoyé et deux valeurs, Startgas et Gasprice. Gasprice est les frais payés pour le mineur par étape de calcul pour l'exécution de la transaction et Startgas est défini pour limiter les étapes de calcul, ce qui empêche l'expansion exponentielle et les boucles infinies dans le code. Si la transaction manque de gaz, tous les changements sont annulés. Les changements d'état sont illustrés dans la **Figure 5(page 14)**. La transaction contient l'adresse de l'expéditeur et du destinataire, la valeur envoyée en Ether, les données supplémentaires envoyées et la signature[14].

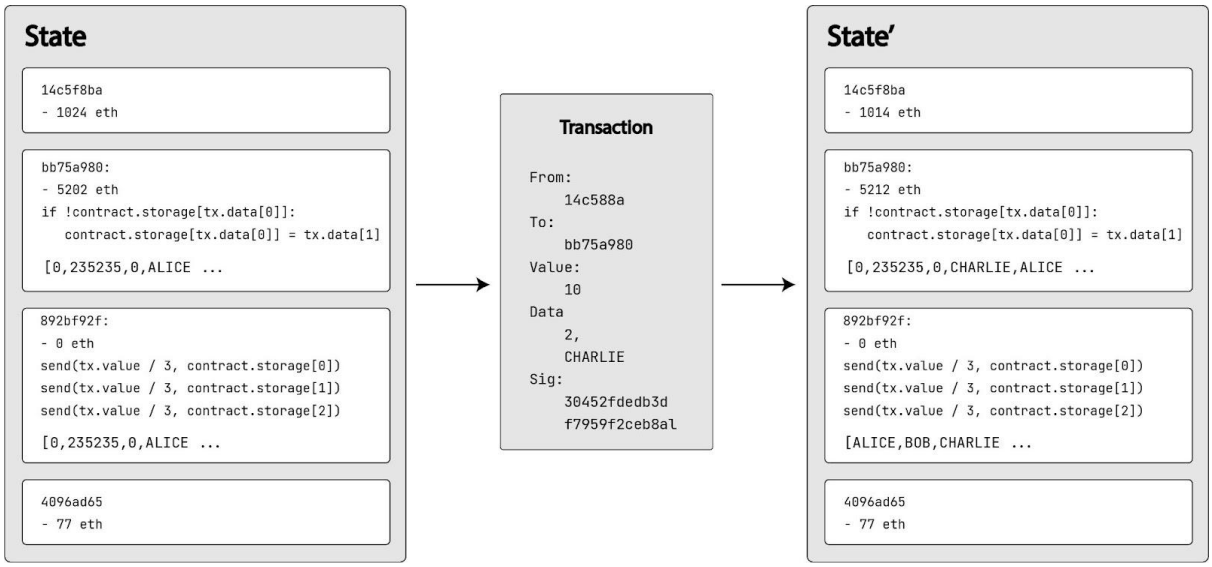

Figure 5 : Ethereum Transaction[14].

1.6 Exemple d'entités qui se définissent comme des DAO

Il existe de nombreux exemples de DAO différents qui ont été créés pour des buts variés. Les plateformes de déploiement de DAO permettent aux utilisateurs de créer leur propre DAO en utilisant un modèle personnalisable. Les plateformes les plus importantes sont Aragon, DAOstack, DAOhaus et Colony. Elles sont toutes des projets gratuits/open-source en développement à différents stades de maturité.

Dans cette section, nous nous concentrerons sur les plateformes de DAO Aragon et DAOstack.

Aragon Dao

Aragon est une plateforme créée en novembre 2016 pour faciliter la création et la gestion de DAO (Organisations autonomes décentralisées) pour les entrepreneurs et les investisseurs. En novembre 2018, le réseau Aragon a été lancé, offrant un tribunal en ligne pour résoudre les différends de manière binaire, avec des résultats limités à "oui" ou "non", "vrai" ou "faux".

Actuellement, Aragon est géré par une association basée en Suisse, qui détient la propriété intellectuelle, les produits de la vente de jetons, les sites web et les médias sociaux, et emploie l'équipe fondatrice. Cela lie Aragon à une personne juridique et à un ordre juridique existant.

Cependant, Aragon cherche à devenir indépendant de cette association sous-jacente et à couper tous les liens avec un ordre juridique. Ils prévoient de transférer tous les actifs au réseau Aragon afin de réaliser leur vision d'autonomie. Une fois ce transfert d'actifs effectué, le statut juridique d'Aragon, du réseau Aragon et de ses participants reste incertain[4].

DAOStack

DAOstack propose une plate-forme innovante pour résoudre le problème de l'échelle de la gouvernance dans les DAO. En utilisant le système de prise de décision Holographic Consensus (HC), les membres de la DAO peuvent voter sur des propositions qui nécessitent une majorité absolue pour être adoptées. Cependant, un marché de

prédiction permet aux membres de parier sur le résultat des propositions, contournant ainsi l'exigence de vote à majorité absolue si un seuil de mise est atteint. Ce système agit comme un filtre pour la communauté, en attirant l'attention sur les propositions qui suscitent l'intérêt des parieurs et en éliminant les propositions moins favorables. Les parieurs sont incités à être alignés sur les opinions globales de la DAO, ce qui permet une meilleure évolutivité pour les grandes communautés de DAO. Des études préliminaires indiquent que le système fonctionne comme prévu.[19].

Les DAOs mentionnés précédemment partagent trois caractéristiques communes : ils sont des actifs organisés selon des règles de gouvernance, ils utilisent des contrats intelligents fonctionnant sur la blockchain Ethereum pour leur fonctionnement, et ils sont décentralisés et distribués, avec un contrôle partagé entre tous les participants et un protocole distribué dans un système pair à pair.[5].

1.7 Conclusion

Dans ce chapitre nous avons couvert une variété de sujets relatifs à la technologie sous-jacente aux DAO et aux blockchains. Nous avons examiné en détail les caractéristiques, la structure et les types de blockchains, ainsi que les mécanismes de fonctionnement des smart contracts et la confidentialité dans diverses plateformes de blockchain. Nous avons également exploré les organisations décentralisées autonomes (DAO) en général, en commençant par leurs définitions. Nous avons également étudié les aspects techniques et les processus de prise de décision dans les DAO, en examinant des exemples d'entités qui peuvent être définies comme DAO.

Chapitre 2

Confidentialité dans Les organisation décentralisé

2.1 Introduction

Dans ce chapitre, nous allons approfondir notre compréhension des DAO. Nous allons étudier l'intérêt et le fonctionnement des DAO, ainsi que les problèmes de confidentialité qui peuvent survenir dans ces organisations autonomes décentralisées.

Nous allons prendre pour exemple la ConstitutionDAO pour étudier les problèmes de confidentialité. Pour faire face à ces problèmes, nous explorerons ensuite différentes techniques de confidentialité dans les blockchains. Parmi celles-ci, nous étudierons le concept de commitment, qui permet de cacher certaines informations. Nous aborderons également les preuves à divulgation nulle de connaissance et en particulier le ZK-SNARK (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge), une technique puissante permettant de prouver la connaissance d'une information sans la révéler.

2.2 Comment fonctionne les organisations décentralisées

Les membres d'une DAO sont généralement représentés par un actif numérique. Ces "jetons de gouvernance" permettent aux détenteurs de proposer et de voter sur des modifications du protocole. Les propositions peuvent aller de mises à niveau de la cybersécurité à des révisions de la mission de l'organisation. Bien que certaines DAO soient privées, la plupart sont basées sur des actifs numériques négociables librement, permettant à tout utilisateur d'obtenir des jetons de gouvernance et de faire partie de la DAO. Les DAO peuvent également accorder des allocations initiales, notamment par des airdrops - dans lesquels des jetons sont fournis gratuitement en échange d'une utilisation antérieure ou en échange d'un service - aux fondateurs et autres parties prenantes qui ont démontré leur engagement avec une plateforme pertinente[20].

Pour établir une DAO, le premier pas critique est de galvaniser une communauté unie par un but commun. Souvent, les DAOs sont lancées par des pairs qui coordonnent leurs activités sur des plateformes de communication telles que Discord, Telegram et Twitter. Les fondateurs travaillent ensemble pour déterminer le but de la DAO, s'entendre sur les paramètres de gouvernance et élaborer un plan de lancement. Les communautés DAO utilisent souvent des symboles, des mèmes, des acronymes et d'autres références pour s'organiser. Une DAO peut également être construite autour d'une communauté existante ou d'une application basée sur la blockchain[20].

Une fois que le groupe a atteint un accord, la communauté peut alors encoder son mandat et ses règles dans des contrats intelligents, qui lieront finalement le groupe à ses décisions. Bien que certaines DAOs optent pour coder leurs propres règles, les plateformes de création de DAO telles que Gnosis, Aragon, Colony et DAOStack offrent des outils prêts à l'emploi pour développer le code de contrats intelligents. Les utilisateurs de ces services peuvent définir des paramètres tels que le jeton principal, la vitesse des propositions, la période de vote, les mécanismes de vote et les mécanismes de proposition[20].

2.3 Problème de transparence sur les organisations décentralisée

2.3.1 L’exemple de ConstitutionDAO

Un DAO est une forme de partenariat où un groupe de personnes se réunit pour atteindre un objectif commun. Récemment, un DAO appelé ConstitutionDAO s’est formé avec la mission d’acheter l’une des dernières copies physiques restantes de la Constitution. La DAO a levé la somme étonnante de 40 millions de dollars en moins d’une semaine pour faire une offre. Cependant, la DAO a été rapidement surenchérit par un investisseur individuel qui a placé une offre de 43,17 millions de dollars, juste au-dessus des actifs de ConsutitutionDAO, assurant ainsi que l’investisseur a remporté l’enchère[21].

2.3.2 Problèmes de la trésorerie publique dans les DAO

C’est là que réside le problème avec l’état actuel des DAOs : leurs trésoreries sont publiques. Lorsque ConstitutionDAO levait des fonds, tout son bilan était visible pour le

public. Cette propriété peut nuire aux DAO et nuire à leur capacité de participer à certains types d'enchères. En particulier, les DAOs ne peuvent pas participer aux enchères scellées précisément parce que leurs trésoreries sont publiques. Dans le cas de ConstitutionDAO, le fait d'avoir un trésor public permettait aux autres soumissionnaires de connaître son offre maximale et de surenchérir.[21].

2.4 Techniques de confidentialité utilisées dans la blockchain

Il existe diverses techniques qui peuvent être mises en œuvre pour renforcer la sécurité et la confidentialité des systèmes blockchain. Parmi ces techniques, nous avons sélectionné trois techniques spécifiques pour fournir une explication détaillée et concrète.

2.4.1 Ring signature

Le Ring Signature est une technique cryptographique qui peut assurer l'anonymat en permettant à n'importe quel membre d'un groupe de signer un message sans que sa véritable identité ne soit révélée. Elle est différente des groupes de signatures, qui ont un gestionnaire de groupe et ne peuvent pas être créés facilement par les utilisateurs. Le Ring Signature est utilisé dans CryptoNote pour cacher les adresses des expéditeurs des transactions. Ethereum a également ajouté la fonctionnalité de Ring Signature en 2015 pour offrir l'anonymat aux utilisateurs comme les cryptomonnaies CryptoNote telles que Monero. Le nombre de membres de l'anneau détermine la probabilité qu'un adversaire puisse deviner l'identité du véritable expéditeur d'une transaction[23].

2.4.2 Preuve de connaissance nulle interactive (NIZK)

Une autre technologie cryptographique offrant des propriétés de préservation de la vie privée est les preuves à divulgation nulle (zero-knowledge proofs), proposée dans les années 1980. L'idée de base est qu'une preuve formelle peut être formulée pour vérifier qu'un programme exécuté avec une entrée connue de manière privée par l'utilisateur peut produire une sortie ouverte publiquement sans divulguer d'autres informations. Dans une application de blockchain, tous les soldes de compte sont cryptés et stockés dans la chaîne. Lorsqu'un utilisateur transfère de l'argent à un autre utilisateur, il peut facilement prouver

qu'il dispose d'un solde suffisant pour le transfert avec des preuves à divulgation nulle, sans révéler le solde du compte. Une variation de cette technologie est la preuve à divulgation nulle succincte et non interactive (zk-SNARK), qui est utilisée dans le protocole Zcash pour vérifier les transactions tout en protégeant la vie privée des utilisateurs. Le groupe Zcash a récemment amélioré le langage de contrat Ethereum pour fournir efficacement la vérification de preuves zk-SNARK. Ainsi, cette technologie permet de garantir la confidentialité des données stockées sur la blockchain[23].

2.5 Atteindre la confidentialité sur une blockchain publique

Le problème fondamental que nous allons examiner est le suivant : peut-on avoir des transactions privées sur une blockchain publique ? Rappelons qu'une des propriétés de base d'une blockchain est la vérifiabilité universelle, ce qui signifie que n'importe qui peut vérifier que les règles d'une blockchain sont respectées et que seules les transactions valides sont inscrites dans le registre des transactions. À première vue, on pourrait penser que si nous voulons soutenir la vérifiabilité universelle, toutes les données de transaction doivent être publiques, sinon, comment vérifier que les transactions sont valides ? Cependant, il s'avère que ce raisonnement est incorrect et que nous pouvons en réalité avoir des transactions privées sur une blockchain publique avec une vérifiabilité universelle. Nous parvenons à cela en utilisant des outils de cryptographie que nous présenterons ultérieurement. Ces outils cryptographiques utilisent principalement ce que l'on appelle des preuves de connaissance nulle associées à des engagements.

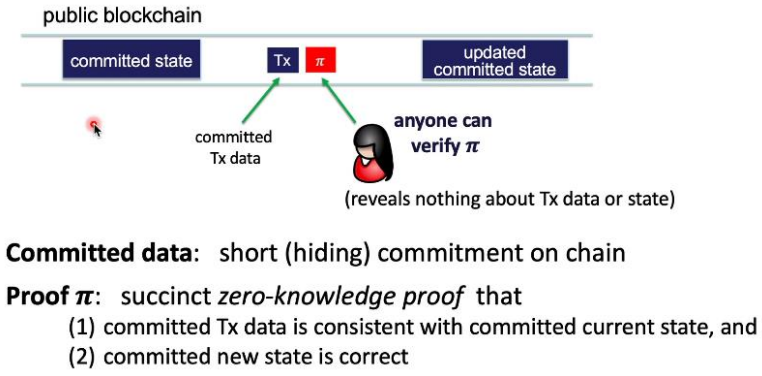

Figure 6 : Transaction Tx privée avec vérifiabilité universelle.

La manière dont nous parvenons à des transactions privées avec une vérifiabilité universelle sur une blockchain publique consiste essentiellement à tout engager plutôt qu'à tout rendre visible sur la blockchain. Par exemple, la blockchain ne contiendra qu'un engagement envers l'état actuel de la blockchain, ce que l'on appelle un engagement caché, qui ne révèle rien sur l'état réel de la chaîne. Ces engagements sont très courts, de sorte que ce qui est réellement écrit est une chaîne très courte, peut-être seulement de 32 octets, en tant qu'engagement caché de l'état. L'état ne peut pas être modifié, mais le public ne peut pas réellement savoir quel est l'état. Lorsqu'une transaction est publiée, nous ne publierons qu'un engagement caché des données de la transaction, puis nous publierons un engagement caché de l'état de la blockchain mis à jour.

Dans ce cas, si tout est engagé et rien n'est visible, la question suivante est : comment pouvons-nous vérifier que c'est une transaction valide ? C'est précisément le rôle de cette preuve (π), ce que l'on appelle une preuve de connaissance nulle succincte. Il s'agit d'une preuve très courte et rapide à vérifier. Cette preuve démontre que les données de transaction engagées sont effectivement cohérentes avec l'état courant engagé. En d'autres termes, la transaction engagée est une transaction valide et l'état engagé mis à jour est le résultat correct de l'application de la transaction à l'ancien état engagé. L'aspect intéressant est que la preuve ne révèle rien sur les données de transaction ou sur l'état.

2.6 Types de confidentialité

Il existe de nombreux types de confidentialité dans les différents systèmes existants, mais nous allons nous concentrer sur deux d'entre eux:

- **Pseudonymat**: il est souvent appelé confidentialité faible, où chaque utilisateur dispose d'un pseudonyme et toutes les transactions de l'utilisateur sont liées à ce pseudonyme. Par exemple, sur Reddit, tous les messages peuvent être liés à un pseudonyme, mais nous supposons que les gens ne peuvent pas relier ce pseudonyme à l'entité physique qui le contrôle. L'avantage des systèmes basés sur des pseudonymes est qu'il est très facile de mettre en place une réputation. Cependant, il existe un risque pour la confidentialité : si l'un de ces messages est lié à l'initiateur, alors tous les

messages sont essentiellement liés à lui, il est donc très difficile de maintenir la séparation entre l'entité physique et le pseudonyme. C'est pourquoi les systèmes basés sur des pseudonymes sont généralement considérés comme offrant une confidentialité faible.

- **Anonymat complet**: toutes les transactions sont indiscernables, ce qui signifie que même si une entité publie deux transactions, personne ne peut savoir que ces deux transactions proviennent de cette même entité. Il est donc beaucoup plus difficile de maintenir une réputation, car il n'y a aucun enregistrement des actions de l'entité dans le temps. Cependant, des systèmes de réputation dans un contexte d'anonymat complet existent, mais ils sont beaucoup plus complexes qu'un simple système basé sur un pseudonyme.

Maintenant que nous comprenons les différents niveaux de confidentialité, examinons le type de confidentialité offert par les plateformes les plus réputées qui existent:

Ethereum: L'unité de base dans Ethereum est un compte, où chaque compte possède une adresse publique qui peut révéler son solde, et toutes les transactions associées à cette adresse particulière sont effectivement liées les unes aux autres. Ainsi, l'adresse agit en réalité comme un pseudonyme, où toutes les transactions effectuées sur cette adresse sont essentiellement liées à ce pseudonyme. Ethereum est en quelque sorte un exemple classique d'un système basé sur un pseudonyme, et il est complexe de s'assurer qu'il existe une séparation physique entre l'adresse du compte et l'entité physique qui en est propriétaire.

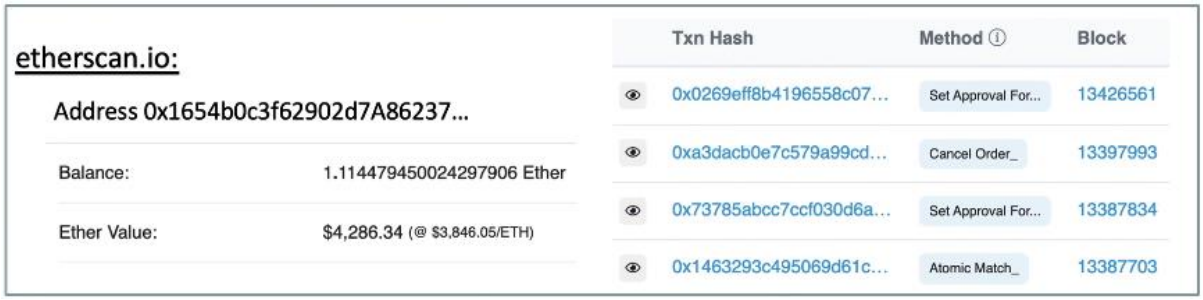

Figure 7 : Pseudonymat sur les comptes Ethereum.

Bitcoin: Chaque transaction contient des informations telles que les adresses d'envoi, les montants, les adresses de réception et les montants reçus par ces adresses. Ainsi, dans le cas de Bitcoin, nous avons également une forme de pseudo anonymat où ces adresses appartiennent à une entité unique, mais la connexion peut ne pas être évidente. Néanmoins, si l'on examine les journaux de transactions sur Bitcoin, on peut constater que certaines entités possèdent plus d'une adresse. En analysant les données de transaction, il est possible de construire progressivement un graphique de toutes les transactions et d'identifier très rapidement toutes les adresses appartenant à un seul utilisateur. Si l'une de ces adresses effectue une transaction avec une entité physique qui connaît réellement l'entité physique associée à l'adresse, alors toutes les adresses deviennent connectées à une seule entité physique. En réalité, des entreprises telles que Chainalysis fournissent des services permettant de relier les adresses Bitcoin à des entités physiques.

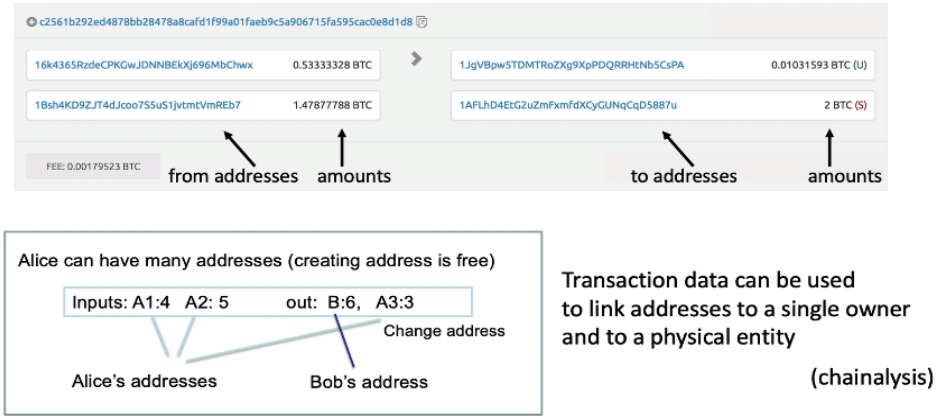

Figure 8 : Pseudonymat sur les transactions Bitcoin.

En général, nous pouvons constater qu'il existe différents niveaux de confidentialité offerts par le système financier, comme le montre le schéma ci-dessous. Par exemple, sur Bitcoin et Ethereum, les transactions sont publiquement visibles et peuvent être liées à au moins un pseudonyme. Dans le cas des systèmes financiers traditionnels tels que le système de cartes de crédit, les paiements sont visibles par la banque mais moins par le monde extérieur, ce qui constitue un niveau de confidentialité différent. Par la suite, il s'avère qu'en utilisant la technologie blockchain, nous pouvons construire un système entièrement privé en utilisant des systèmes comme Turnado Cash et Zcash, où même les opérateurs du système ne sauront pas qui effectue des transactions avec qui.

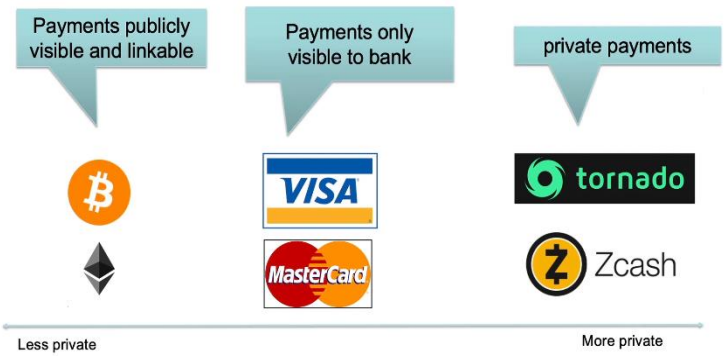

Figure 9 : Différents niveaux de confidentialité dans les paiements numériques.

D'autre part, un exemple simple de mécanisme d'anonymat sur la blockchain est l'utilisation d'un mélangeur (Mixer) (figure 10). Imaginons que nous trouvions un mélangeur qui offre un service de mélange et qu'il soit situé à l'adresse M. Les utilisateurs Alice, Bob et Carol envoient chacun 1 ETH au mélangeur et enregistrent cette transaction sur la blockchain. Ensuite, ils enverront en privé trois nouvelles adresses (X, Y, Z) à ce mélangeur, et lui seul saura à qui appartient chaque adresse. Le mélangeur enverra alors 1 ETH à chacune de ces adresses. Maintenant, pour un observateur qui regarde la blockchain, il sait par exemple que l'adresse Y appartient à l'un des utilisateurs Alice, Bob et Carol, mais il ne sait pas lequel possède réellement cette adresse Y. Nous disons donc que l'ensemble d'anonymat est de taille 03. Bien sûr, nous pourrions mélanger ces adresses avec des centaines d'entités pour obtenir un ensemble d'anonymat plus important, ou bien prendre l'adresse Y de Bob et la mélanger avec d'autres parties, ce qui augmentera rapidement l'ensemble d'anonymat.

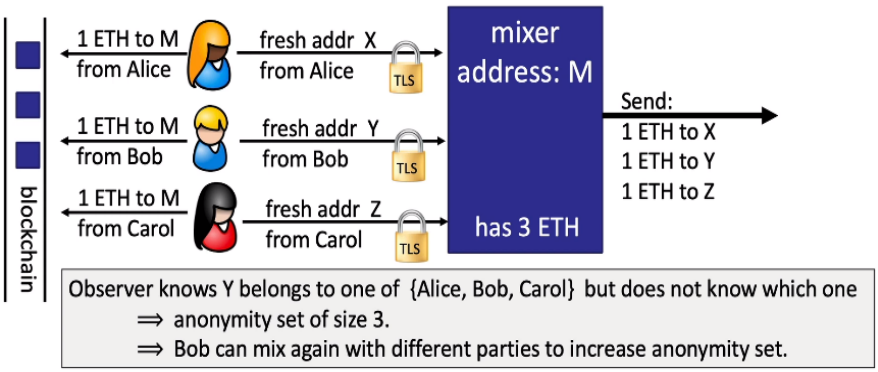

Figure 10 : Simple Anonymat via un mélangeur (Mixing).

C'est ainsi qu'un mélangeur très simple parvient à assurer l'anonymat, mais il existe évidemment certains problèmes avec cette conception. Tout d'abord, le mélangeur connaît la correspondance entre les adresses de sortie et les adresses d'entrée, alors que le premier objectif était de construire un système où personne ne connaît cette correspondance. De plus, le mélangeur pourrait simplement disparaître avec les Ether qui lui ont été confiés et ne pas payer les parties concernées, ce qui est un problème beaucoup plus sérieux.

Cependant, il existe des solutions à ce problème qui permettent de résoudre la question du mélange sans un mélangeur de confiance. Sur Bitcoin, il existe ce qu'on appelle le protocole CoinJoin, qui est mis en œuvre sur le portefeuille Wasabi, et sur Ethereum, il y a Tornado Cash, qui est un mélangeur sans confiance. Ces solutions visent à fournir un processus de mélange sans confiance dans lequel les utilisateurs peuvent combiner leurs transactions avec celles des autres de manière à rendre difficile la traçabilité des entrées individuelles vers leurs sorties respectives. Le protocole CoinJoin permet à plusieurs participants de créer collaborativement une transaction en combinant leurs entrées et leurs sorties, ce qui rend difficile l'association d'entrées spécifiques à des sorties spécifiques. Tornado Cash, quant à lui, utilise des contrats intelligents et des preuves à divulgation nulle (zero-knowledge proofs) pour permettre le mélange privé et sans confiance de l'Ether.

2.7 Engagements cryptographiques (Cryptographic commitments)

Un engagement cryptographique émule ce que nous appelons dans le monde physique une enveloppe (figure 11), où Bob peut placer des données à l'intérieur d'une enveloppe qui cache la nature de ces données et l'envoyer à Alice. Du fait que les données se trouvent à l'intérieur de l'enveloppe, Alice ne sait pas réellement ce qu'il y a à l'intérieur. Cependant, plus tard, Bob peut simplement ouvrir l'enveloppe et convaincre Alice qu'il s'agit bel et bien des données initialement prévues.

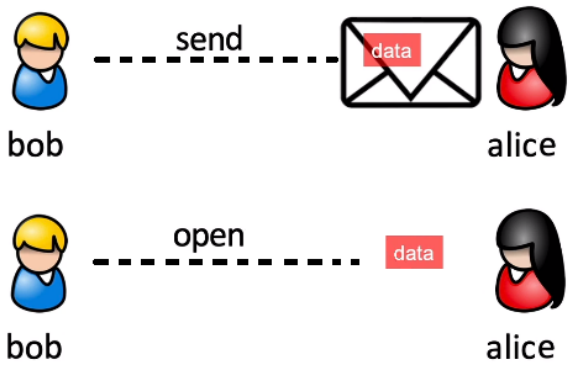

Figure 11 : Engagement cryptographique.

2.7.1 Schéma de commitment (Commitment scheme)

La syntaxe d'un schéma de commitment se compose essentiellement de deux algorithmes :

- $Commit(msg, r) \rightarrow com$
- $Verify(msg, com, r) \rightarrow \{accepter\ rejeter\}$

msg : Le message auquel nous nous engageons.

r : Une variable aléatoire secrète dans un espace aléatoire R.

com : La chaîne de commitment.

L'algorithme de commitment prend en entrée des données auxquelles s'engager, appelées message msg, et une variable aléatoire r sélectionnée dans un espace de variables aléatoires R, produisant ainsi la chaîne de commitment com. Cette chaîne est la valeur qui peut être envoyée à Alice. Ultérieurement, lorsque Bob souhaite ouvrir le commitment, il peut envoyer le message et la variable aléatoire r à Alice, qui peut exécuter l'algorithme verify afin de s'assurer que le message est réellement une ouverture de l'engagement com par rapport à la variable aléatoire r. Une fois que le commitment est ouvert, n'importe qui peut vérifier que l'engagement a effectivement été ouvert correctement.

2.7.2 Propriétés de sécurité d'un schéma de commitment

Il existe deux propriétés de sécurité importantes pour les schémas de commitment:

A. **Contraignant / Engagement (Binding) :**

Cette propriété garantit que, une fois que l'expéditeur s'engage à certaines données, il ne peut plus changer d'avis quant aux données à l'intérieur du commitment. Cela signifie que :

$$\forall com \nexists (m1, r1), (m2, r2) : \{verify(m1, com, r1) = verify(m2, com, r2) = accepter\ m1 \neq m2$$

Plus précisément, l'expéditeur n'est pas en mesure de produire une seule chaîne de commitment qui peut être ouverte de deux manières différentes, et une fois fixée, celui qui s'engage ne peut plus modifier les données engagées.

B. **Dissimulation / Caché (Hiding) :**

Cette propriété garantit que le commitment ne révèle rien sur les données engagées. Ainsi, la chaîne de commitment com, qui provient d'un message m et d'un échantillonnage aléatoire de la chaîne r, est statistiquement indépendante de m ou ce qui est autrement appelé "engagement caché sans condition".

2.7.3 Exemples de construction d'un schéma de commitment

Voyons maintenant quelques exemples de construction de schémas de commitment:

Example 01 : Commitment basé sur le hachage (Hash-based Commitment) :

En fixant une fonction de hachage : $H: M\ x\ R \rightarrow C\ \ (e.g\ SHA256)$

$$où\ H\ est\ résistante\ aux\ collisions\ et\ |R| \gg |C|$$

- $commit(m \in M, r \leftarrow R)$: $com = H(m, r)$
- $verify(m, com, r)$: $accepter\ si\ com = H(m, r)$

Dans ce schéma de commitment, nous hachons un message avec une chaîne aléatoire, ce qui nous donne une valeur de commitment aléatoire qui engage celui qui prend

l'engagement au message m. Ce schéma est évidemment contraignant en raison de la propriété de résistance aux collisions de la fonction de hachage H, et il est également dissimulé car com est un hachage du message (m, r).

Example 02 : Commitment de Pedersen (Pedersen Commitment)

Ce schéma de commitment provient d'une fonction de hachage algébrique définie comme suit :

G est un groupe cyclique fini défini comme suit :

$$G = \{1, g^1, g^2, \dots, g^{q-1}\} \quad où \quad g^i . g^j = g^{(i+j \bmod q)}$$

Où g est appelé le générateur, et $q = |G|$ est appelé l'ordre de G ou le nombre d'éléments dans le groupe G, et on suppose que q est un nombre premier.

En fixant

$$g, h \; dans \; G \quad \text{et} \quad R = \{0,1, \dots, q-1\}$$

Pour $(m, r) \in R$ on définit la fonction de hachage $H(m, r) = g^m . h^r \in G$

Le schéma d'engagement est défini par :

$$commit(m \in R, r \leftarrow R) = H(m, r) = g^m . h^r$$

Ce schéma d'engagement est contraignant et dissimulé pour la même raison que l'exemple précédent. Mais il présente également une propriété particulière appelée propriété homomorphe, ce qui signifie que, étant donné deux engagements com_1 et com_2 où le premier est un engagement envers m_1 et le second envers m_2:

$$commit(m_1 \in R, r_1 \leftarrow R) \rightarrow com_1$$
$$commit(m_2 \in R, r_2 \leftarrow R) \rightarrow com_2$$

La propriété remarquable est que si on multiplie ces deux engagements.

$$com_1 . \; com_2 = g^{m1+m2} . h^{r1+r2} = commit(m_1 + m_2, r_1 + r_2)$$

On obtient miraculeusement l'engagement de la somme de m_1 and m_2 en utilisant le paramètre aléatoire $r_1 + r_2$.

Ce qui est intéressant avec cette propriété, c'est que n'importe qui peut prendre deux chaînes d'engagement sans avoir aucune idée des données engagées, mais en les multipliant, ils peuvent construire l'engagement de la somme des valeurs engagées même s'ils ne connaissent pas réellement les valeurs originales.

2.8 Zero-knowledge succinct non-interactive argument of knowledge

ZK-SNARK ou argument succinct non interactif de connaissance (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) possède un nombre considérable d'applications dans le domaine de la blockchain, en particulier lorsqu'il est question de la confidentialité des données de transaction. Ils sont extrêmement utiles également lors de la création de systèmes offrant des services de confidentialité tels que Tornado Cash, Zcash, voire la maintenance du code des contrats intelligents caché afin que personne ne puisse voir la logique commerciale d'une entreprise qui tente de déployer un contrat intelligent particulier. En réalité, il existe une entreprise appelée Aleo qui permet de déployer des Dapps sur une blockchain sans révéler réellement le code qui les sous-tend. Les zk-SNARKs sont également utiles dans des applications de conformité. Par exemple, une plateforme d'échange pourrait vouloir prouver qu'elle est solvable de manière indiscernable, ce qui signifie que cette plateforme peut prouver qu'elle détient plus d'actifs que d'obligations sans révéler aucune donnée sensible de l'entreprise au public.

2.8.1 Circuits arithmétiques (Arithmetic Circuits)

Pour expliquer ce qu'est un zk-SNARK, nous devons d'abord expliquer ce qu'est un circuit arithmétique ; fixons un certain champ fini:

$$F = \{0, \dots, p-1\} \quad pour\ un\ nombre\ premier \quad p > 2$$

Le champ fini F est un ensemble d'entiers allant de 0 à $p-1$, où toutes les opérations arithmétiques sont effectuées modulo p. Le circuit arithmétique est défini par le graphe acyclique dirigé (DAG, Directed Acyclic Graph) C où

$$C: F^n \rightarrow F$$

- Les nœuds internes sont étiquetés $+, -$ or x
- Les entrées sont étiquetées $1, x_1, \ldots, x_n$

Ainsi, un circuit arithmétique prend en entrée une liste de n éléments dans le champ et produit un élément dans ce même champ.

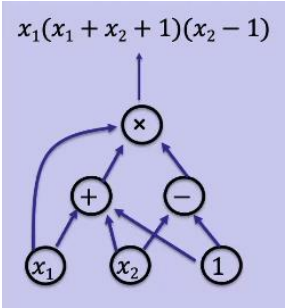

Figure 12 : Exemple d'un circuit arithmétique.

Comme le montre l'exemple, un circuit arithmétique définit un polynôme multivarié ainsi qu'une manière particulière d'évaluer sa valeur en fonction des entrées du graphe. On note $|C|$ la taille du circuit, qui fait référence au nombre de portes arithmétiques (arithmetic gates) dans le circuit.

$$|C| = \#portes\ dans\ C$$

Dans notre exemple, nous disons que le circuit arithmétique a une taille de 3 car il contient 3 portes arithmétiques.

Les circuits arithmétiques sont un modèle de calcul très puissant qui nous permet d'effectuer différents calculs. Voici un exemple utilisant la fonction de hachage SHA256 :

2.8.2 Circuit de hachage (Hashing Circuit)

Ce circuit prend en entrée un hachage h et un message m, et il produit en sortie 0 si h est le résultat du hachage de ce message en utilisant l'algorithme SHA256.

$$C_{hash}(h,m):\{0 \quad si \quad SHA256(m) = h \qquad \neq 0 \quad sinon$$

Nous pouvons réécrire le circuit de la manière suivante:

$$C_{hash}(h,m) = h - SHA256(m)$$

La mise en œuvre de ce circuit nécessitera environ 20 000 portes en raison de la définition spécifique de la fonction de hachage SHA256. Ce circuit n'est pas un circuit particulièrement petit, mais ce n'est pas non plus un circuit très volumineux.

$$|C_{hash}| \approx 20k\ portes$$

2.8.3 Systèmes d'arguments (Argument systems)

Les systèmes d'arguments impliquent deux parties, le Prouveur (Prover) et le Vérificateur (Verifier), souvent notés P et V. En fixant un circuit particulier C, qui prend deux entrées x, w, où $x \in F^n$ et $w \in F^m$ sont des listes respectivement composées de n et m éléments du champ. On va désigner x comme l'énoncé que nous essayons de prouver et w comme un témoin secret que le Prouveur essaie de démontrer sa connaissance au vérificateur.

Dans cette configuration, le Prouveur reçoit l'énoncé x et le témoin w, tandis que le Vérificateur ne reçoit que l'énoncé x. Ce que le Prouveur essaie de faire est de convaincre le Vérificateur qu'un témoin w existe tel que:

$$C(x,w) = 0$$

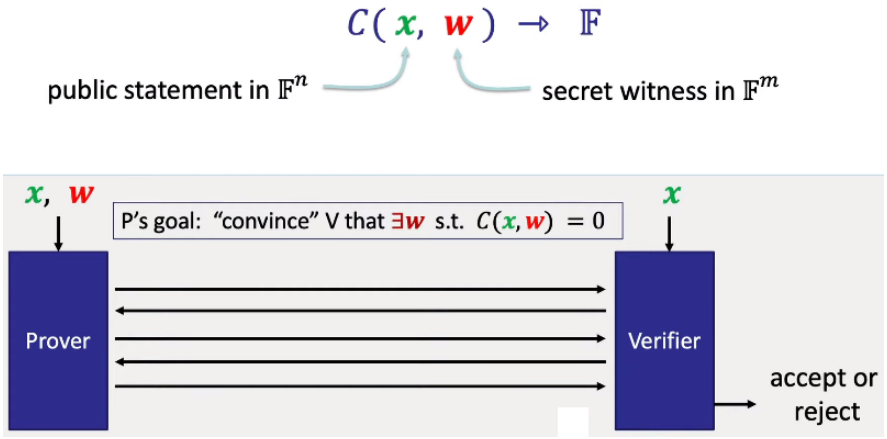

Figure 13 : Système d'argumentation interactif.

La manière dont cela sera fait est en s'envoyant mutuellement des messages, où le Vérificateur défie le Prouveur et vérifie ses réponses. En se basant sur celles-ci, il décidera enfin d'accepter ou de rejeter la preuve.

Il est important de comprendre qu'il existe deux types de systèmes d'argumentation :

A. Système d'argumentation interactif

Dans un système d'argumentation interactif le Prouveur et le Vérificateur peuvent avoir de nombreuses interactions en s'envoyant mutuellement des messages. Finalement, lorsque les échanges sont terminés, le Vérificateur décide d'accepter ou de rejeter la preuve. Ce type de système est très utile lorsqu'il y a un seul Vérificateur, tel qu'un auditeur de conformité à qui nous devons prouver quelque chose, et dans ce cas, il est acceptable d'avoir un protocole interactif.

B. Système d'argumentation non interactif

Dans ce type de système, le Prouveur envoie un seul message au Vérificateur, appelé la Preuve, et le Vérificateur doit décider s'il est convaincu ou non, en se basant sur ce seul message. Ce type de système est particulièrement utile lorsqu'il y a plusieurs Vérificateurs impliqués. Il est important de mentionner que la preuve doit être très courte, car elle doit être publiée sur la blockchain et vérifiée très rapidement, de sorte que les mineurs qui la vérifient n'aient pas à travailler très dur. Si on a une preuve courte qui peut être vérifiée rapidement, nous appelons le système d'argumentation un SNARK ou un argument succinct non interactif de connaissance.

Dans l'environnement de la blockchain, les systèmes d'argumentation non interactifs sont les plus pertinents. C'est pourquoi nous allons nous concentrer sur ce type de systèmes, plus précisément sur les systèmes d'argumentation prétraités.

2.8.4 Système d'argument prétraité

Dans le système d'argument prétraité (preprocessing argument system), le circuit C est tout d'abord prétraité à l'aide d'un algorithme de configuration $S(C)$ qui prend le circuit en entrée et produit deux paramètres publics (S_p, S_v), S_p pour le Prouveur et S_v pour le Vérificateur. Dans ces systèmes d'argument, le Prouveur prend S_p, x, w et le Vérificateur

prend S_v, x sans connaître, comme d'habitude, le témoin w. Le Prouveur envoie ensuite la preuve au Vérificateur, qui utilise la preuve π, S_v, x pour décider d'accepter ou de rejeter la preuve (**figure 14).**

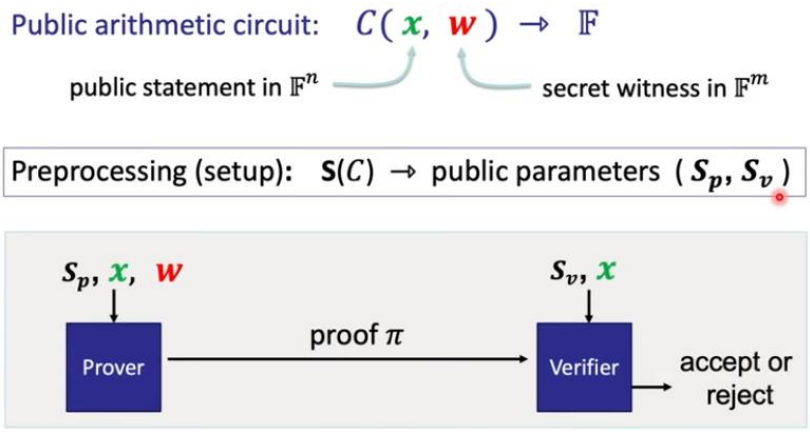

Figure 14 : Système d'argument prétraité non interactif.

Donc, plus en détail, un système d'argument prétraité est un triplet d'algorithmes (S, P, V) où :

- $S(C) \longmapsto paramètres\ publics\ (S_p, S_v)\ pour\ le\ prouveur\ et\ le\ vérificateur$
- $P(S_p, x, w) \longmapsto preuve\ \pi$
- $V(S_v, x, \pi) \longmapsto \{accepter\ rejeter$

Fondamentalement, un système d'argument prétraité doit satisfaire un certain nombre d'exigences, à savoir :

A. Complétude (Completeness)

Cette exigence stipule que si w est un témoin correct pour x, alors le Prouveur honnête convaincra toujours le Vérificateur honnête d'accepter la preuve.

$$\forall\, x, w : C(x, w) = 0 \implies Pr\left[V\left(S_v, x, P(S_p, x, w)\right) = accepter \right] = 1$$

B. Argument de connaissance / Validité : (Argument of knowledge / Soundness)

Cette propriété vérifie que le système d'argument est un argument de connaissance, en d'autres termes, si le Vérificateur accepte la preuve, alors le Prouveur doit réellement connaître le témoin w.

$$V\ accepte\ la\ preuve\ \Longrightarrow\ P\ connaît\ w\ \ t.q\ \ C(x,w)=0$$

Ainsi, si un Prouveur malveillant qui ne connaît pas le témoin w tente de convaincre le Vérificateur du contraire, il a une probabilité négligeable de le faire.

$$P^*\ ne\ connaît\ pas\ w\ \Longrightarrow\ Pr[\ V(S_v,x,\pi)=accepter]<\varepsilon$$

C. Zéro connaissance (facultatif) :

Cette propriété signifie que la preuve (S_v,x,π) ne révèle aucune information sur le témoin w.

D. Succinct :

En plus des trois propriétés mentionnées précédemment, on peut exiger une autre propriété du système d'argument prétraité, à savoir que le système d'argument doit être succinct, ce qui signifie que le Prouveur génère des preuves courtes et que le Vérificateur peut effectuer ses vérifications en très peu de temps.

- $P\big(S_p,x,w\big)\ \longmapsto preuve\ courte\ \pi \qquad |\pi|=O(log\ (|C|)\ ,\lambda)$
- $V(S_v,x,\pi)\ \longmapsto accepter\ où\ rejeter \qquad temps(V)=O(|x|,log\ (|C|)\ ,\lambda)$

On dit que la preuve est courte si sa longueur a effectivement une complexité logarithmique par rapport à la taille du circuit C et à un certain paramètre de sécurité λ, même pour des circuits importants contenant des millions de portes. De plus, on souhaite que la vérification soit très rapide, éventuellement en un temps logarithmique par rapport à la taille du circuit, et en un temps linéaire par rapport à la taille de l'énoncé x, car le vérificateur doit au moins lire l'énoncé qu'il vérifie. Cela explique pourquoi une procédure de prétraitement S est nécessaire pour résumer le circuit C en le sommant, afin d'aider le vérificateur à vérifier des circuits importants dans un temps contraint.

Enfin, si le système d'argument (S,P,V) est à la fois succinct et à connaissance nulle, on dit qu'il s'agit d'un zk-SNARK.

Avant d'aller plus loin, nous devons préciser qu'il existe en réalité plusieurs types de procédures de configuration :

A. Configuration de confiance par circuit (Trusted setup per circuit)

Dans ce type, l'algorithme de configuration utilise des données secrètes dont il est le seul à avoir connaissance. Généralement, ces données secrètes sont simplement des bits aléatoires qu'il génère pour lui-même, mais ces bits aléatoires doivent être gardés secrets et détruits une fois la procédure de configuration achevée son travail, de sorte que personne ne les connaisse après. Dans le cas le plus simple, la procédure de configuration de confiance $S(C)$ doit être exécutée une fois par circuit, de sorte que ces données secrètes soient spécifiques à ce circuit C qui est en cours de traitement. Il est important de se rappeler que si le secret aléatoire utilisé par la procédure de configuration devient public, cela serait désastreux pour l'environnement de la blockchain, car cela permettrait au prouveur de prouver de fausses déclarations, ce qui lui permettrait de générer de l'argent à partir de rien et toutes sortes d'autres problèmes surviendraient. C'est pourquoi nous devons nous assurer que le secret aléatoire utilisé par ce type de procédure de configuration est toujours gardé secret et toujours détruit lorsque le traitement est terminé.

B. **Configuration de confiance universelle (Universal trusted setup)**

Ce type utilise toujours un aléatoire de confiance, mais il est universel pour tous les circuits. En d'autres termes, nous pouvons considérer la procédure de configuration comme étant divisée en deux algorithmes distincts (une initialisation et un prétraitement) :

$$S = (S_{init}, S_{pre}) : \quad S_{init}(\lambda) \rightarrow U, \quad S_{pre}(U, C) \rightarrow (S_p, S_v)$$

L'algorithme S_{init} est exécuté une seule fois générant un aléatoire secret qui doit être détruit une fois son exécution est terminée. Il générera également en résultat une chaîne universelle U. L'algorithme de prétraitement S_{pre} utilise ensuite cette chaîne universelle avec le circuit donné pour générer (S_p, S_v). Cette deuxième partie est entièrement déterministe et tout le monde peut vérifier que ces paramètres ont été construits correctement. Cette approche est

meilleure que le paradigme de configuration précédent, car nous n'avons à exécuter S_{init} qu'une seule fois pour tous les circuits qu'on souhaite prétraiter.

C. Configuration transparente (Transparent setup)

Ce type est l'objectif ultime, où l'algorithme de prétraitement n'a besoin d'aucune donnée secrète du tout. Fondamentalement, il générera (S_p, S_v) uniquement à partir du circuit, et n'importe qui peut vérifier que ces paramètres ont été calculés correctement. Il existe des SNARKs qui utilisent une configuration transparente, mais ils ne sont pas aussi efficaces que ceux qui peuvent utiliser soit la configuration de confiance par circuit, où la configuration de confiance universelle.

Il y a eu des progrès considérables dans la construction des SNARKs. En fait, les premiers SNARKs ont été conçus au début des années 1990 basés sur le théorème PCP (Probabilistically Checkable Proof), on peut citer par exemple **[**Kilian'92**]** et **[Micali'94]**. Ces SNARKs étaient très courtes et transparentes, et avaient des vérifications très rapides. Cependant, le problème était que le temps nécessaire aux prouveurs était si long qu'il serait impraticable de déployer ces SNARKs.

En 2013, une idée importante a été proposée par Gennaro, Gentry, Parno et Raykova **[GGPR'13]**, et elle a été améliorée en 2016 par Groth **[Groth'16]**, qui a construit un SNARK avec un prouveur en temps linéaire et une preuve de taille constante. Cette construction nécessite une configuration de confiance pour chaque circuit. Pour les SNARKs avec une configuration de confiance universelle, nous pouvons citer **[Sonic'19]**, **[Marlin'19]** et **[Plonk'19]**, dans lesquels la configuration de confiance est exécutée une fois et utilisée pour prétraiter autant de circuits qu'on souhaitait. En ce qui concerne les SNARKs sans configuration de confiance (transparente), nous avons également **[DARK'19]**, **[Halo'19]** et **STARK**, mais ceux-ci produisent soit des preuves légèrement plus longues que les autres systèmes, soit le prouveur peut prendre un peu plus de temps à s'exécuter. Le tableau suivant compare une liste partielle des SNARKs existants :

Tableau 1 : Liste partielle des Types de SNARKs

SNARK	Taille de la preuve π	Taille de S_p	Temps de vérification	Configuration de confiance
Groth'16	$O(1)$	$O(\|C\|)$	$O(1)$	Oui/par circuit
Plonk'19 / Marlin'19	$O(1)$	$O(\|C\|)$	$O(1)$	Oui/universelle
Bulletproofs	$O(log\|C\|)$	$O(1)$	$O(\|C\|)$	Non
STARK	$O(log\|C\|)$	$O(1)$	$O(log\|C\|)$	Non
DARK'19	$O(log\|C\|)$	$O(1)$	$O(log\|C\|)$	Non

Veuillez noter que ce tableau fournit une comparaison partielle des SNARKs répertoriés et de leurs caractéristiques.

2.9 Conclusion

Dans ce chapitre, nous avons approfondi notre compréhension des DAO et examiné leur intérêt et leur fonctionnement, ainsi que les problèmes de confidentialité auxquels elles peuvent être confrontées. En nous concentrant sur l'exemple de la ConstitutionDAO, nous avons réalisé l'importance de la confidentialité dans ces entités décentralisées.

Afin de résoudre ces problèmes, plusieurs techniques de confidentialité dans les blockchains ont été explorées, notamment le concept de commitment et les preuves à divulgation nulle de connaissance, tels que le ZK-SNARK. Ces techniques puissantes permettent de masquer des informations sensibles tout en prouvant la connaissance de ces informations sans les révéler.

Chapitre 3

Etude de l’existant

3.1 Introduction

L'un des principaux problèmes des DAO est lié à la visibilité des ressources financières. Lorsqu'une DAO collecte des fonds, que ce soit par le biais d'une vente de jetons, d'une levée de fonds ou d'autres mécanismes, les informations sur les trésoreries deviennent généralement accessibles au public. Cette transparence peut présenter des inconvénients pour les DAO dans certaines situations, notamment lorsqu'elles souhaitent participer à des enchères scellées.

Le principal problème de la trésorerie dans les DAO réside dans la perte de confidentialité des informations financières sensibles. Comme le trésorerie est accessible publiquement sur la blockchain, les détails des transactions et des soldes peuvent être consultés par tous les participants de la blockchain, ce qui peut poser certains problèmes :

- Exposition des informations personnelles : Les transactions effectuées avec les fonds publics peuvent révéler des informations personnelles sensibles, telles que les adresses de portefeuille, les soldes de compte et les habitudes de dépenses. Cela peut compromettre la vie privée des individus et les exposer à des risques potentiels, tels que la surveillance ou le ciblage malveillant.
- Concurrence et fuite d'information : Dans un environnement compétitif, la transparence peut donner un avantage concurrentiel à d'autres projets ou acteurs du marché. Les informations financières accessibles publiquement peuvent être exploitées par des concurrents pour prendre des décisions stratégiques, copier des modèles commerciaux ou nuire à la position du DAO sur le marché.

3.2 Une étude de cas avec ConstitutionDAO

ConstitutionDAO met en lumière les défis liés à la transparence financière des DAO lorsqu'ils participent à des enchères scellées. En particulier, elle met en évidence les conséquences préjudiciables de la divulgation des ressources financières d'un DAO, en se concentrant sur l'expérience de ConstitutionDAO lors de sa tentative d'acquérir un exemplaire original de la Constitution américaine.

La transparence financière de ConstitutionDAO s'est avérée préjudiciable lorsque d'autres enchérisseurs ont eu accès aux informations sur les fonds disponibles pour ConstitutionDAO. Ces informations ont permis aux autres participants de connaître l'offre maximale de ConstitutionDAO, donnant ainsi l'avantage à ceux qui étaient prêts à surenchérir au-delà de cette limite.

En raison de la divulgation des ressources financières, il est devenu difficile pour le DAO de remporter l'acquisition de la Constitution américaine. L'avantage concurrentiel a été perdu, mettant en évidence les limites de la transparence totale dans les enchères scellées.

L'expérience de ConstitutionDAO démontre les défis auxquels sont confrontés les DAO en matière de transparence financière dans les enchères scellées. Il est essentiel de trouver un équilibre entre la transparence et la confidentialité des ressources pour permettre aux DAO de maintenir un avantage concurrentiel dans des situations compétitives comme les enchères scellées. Une réflexion approfondie sur ces questions est nécessaire pour améliorer la participation des DAO dans de tels événements à l'avenir.

3.3 Solution de Griffin Dunaif et Dan Boneh (Private DAO)

Le plan est de déployer un seul contrat principal sur la chaîne Ethereum pour gérer plusieurs DAOs, c'est une approche intéressante pour faciliter la création et la gestion de ces DAO. Ce contrat principal agit comme une plate-forme centrale offrant les fonctionnalités nécessaires pour créer et interagir avec les DAOs.

Chaque DAO créé sur cette plateforme aura un gestionnaire DAO désigné ou un groupe désigné de gestionnaires. Ces gestionnaires sont responsables de la gestion et de la gouvernance du DAO, prenant des décisions telles que l'allocation des fonds et l'approbation des propositions.

La plate-forme permet à tout le monde d'envoyer des fonds à un DAO. Cela signifie que les utilisateurs peuvent contribuer au DAO en envoyant des tokens ou de l'Ether à l'adresse du contrat principal. Ces fonds seront ensuite attribués au DAO correspondant et pourront être utilisés selon les décisions prises par les gestionnaires.

3.4 Solution proposée

Le protocole utilisé pour gérer les DAO sur la plateforme comprend trois étapes clé dans leur cycle de vie :

Création du DAO : Le gestionnaire DAO initie la création d'un nouveau DAO. Cette étape se déroule en dehors de la chaîne Ethereum et ne nécessite pas de transactions sur la blockchain. Le gestionnaire DAO définit les paramètres du DAO, tels que le nom et les règles de gouvernance. Une fois le DAO créé, il est prêt à recevoir des dépôts de fonds.

Dépôt : Un utilisateur qui souhaite contribuer au DAO peut effectuer un dépôt. L'utilisateur envoie ses fonds à l'adresse du contrat principal du protocole PDAO sur la chaîne Ethereum. Lors de cette opération, l'observateur ne peut pas déterminer à quel DAO spécifique les fonds sont destinés. L'adresse du DAO bénéficiaire reste confidentielle. Seul le gestionnaire DAO pourra plus tard accéder à ces fonds lors de l'étape de retrait.

Retrait : Le gestionnaire DAO est le seul à pouvoir retirer des fonds du DAO qu'il gère. Lorsqu'il souhaite effectuer un retrait, le gestionnaire DAO doit fournir une preuve à une tierce partie prouvant que le solde du DAO est supérieur à un certain montant requis. Cette preuve peut être utilisée pour démontrer la capacité du DAO à participer à des activités telles que des enchères scellées. Une fois la preuve fournie et vérifiée, le gestionnaire peut retirer les fonds du DAO en utilisant son autorisation spécifique. Le montant retiré devient public, mais le solde restant du DAO demeure confidentiel.

Ces trois étapes du cycle de vie du DAO dans le protocole permettent de garantir la confidentialité des contributeurs, la protection des fonds et la gestion privée des actifs par le gestionnaire DAO.

Nous allons décrire en détail les trois étapes mentionnées précédemment :

L'étape d'initialisation du contrat dans le protocole PDAO se déroule comme suit :

Déploiement initial : Lorsque le contrat principal du protocole PDAO est déployé pour la première fois sur la chaîne Ethereum, il est configuré avec un arbre de Merkle vide de profondeur d (dans notre exemple, $d = 20$). Un arbre de Merkle est une structure de données hiérarchique où chaque nœud interne est le hachage cryptographique des nœuds enfants. Dans un arbre vide, toutes les feuilles sont nulles.

Stockage des données : Le contrat principal stocke deux informations clés :

- **Le hachage racine** : Il s'agit du hachage cryptographique de la racine de l'arbre de Merkle vide. Ce hachage représente l'état initial du contrat.
- **Le compteur "next"** : Il s'agit d'un compteur initialisé à zéro qui indique l'emplacement de la prochaine feuille vide dans l'arbre. Le compteur "next" est utilisé pour assurer un placement ordonné des dépôts dans l'arbre.
- **Gestion de l'arbre de Merkle :** Lorsqu'un dépôt est effectué dans le contrat principal, le contrat attribue l'emplacement "next" à cette feuille et met à jour le compteur "next" pour pointer vers la prochaine feuille vide disponible. Le dépôt peut être effectué par n'importe quel utilisateur et est associé à un DAO spécifique géré sur la plateforme.

Initialisation d'un nouveau contrat : Lorsque le compteur "next" atteint $2^d - 1$ (où d est la profondeur de l'arbre), cela signifie que toutes les feuilles de l'arbre sont occupées. À ce stade, un nouveau contrat doit être initié pour continuer à recevoir les dépôts. Le processus d'initialisation du nouveau contrat est similaire à l'étape d'initialisation décrite précédemment.

L'initialisation du contrat dans le protocole PDAO garantit un stockage ordonné et efficace des dépôts dans l'arbre de Merkle. Cela permet de préserver la confidentialité des contributeurs en ne révélant pas l'emplacement spécifique de leurs dépôts dans l'arbre.

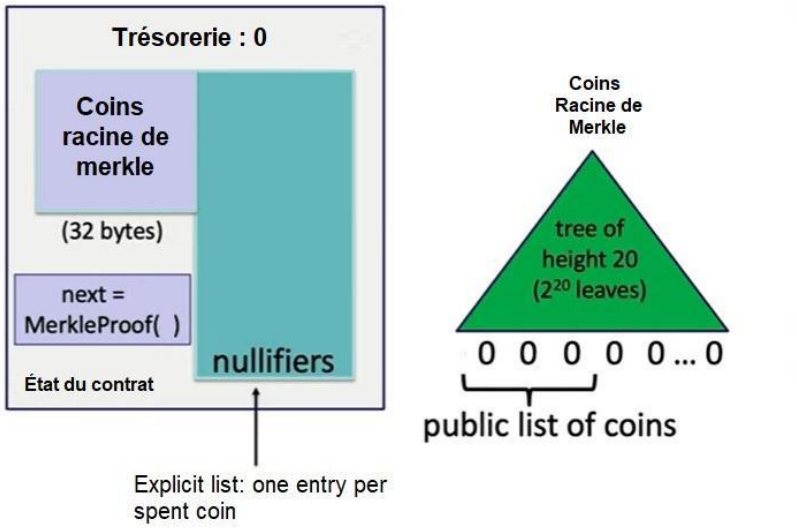

Figure 15 : l'état initial du contrat Treasury.

Étape 1 : Création d'un DAO

Le protocole utilise un groupe cyclique fini E d'ordre premier q avec un générateur $G \in \mathbb{E}$. Ce groupe est basé sur la courbe *elliptique Grumpkin*[24], conçue pour une génération efficace de preuves SNARK sur le réseau Ethereum. La courbe est définie sur un champ premier $\mathbb{F}_r$, où r est la taille de la courbe alt_bn128 utilisée par la précompilation des pairages Ethereum[25]. Cela permet de réaliser efficacement les opérations arithmétiques sur $\mathbb{F}_r$ à l'aide d'un prouveur SNARK.

Maintenant, pour créer un DAO, le gestionnaire du DAO procède comme suit :

- Le gestionnaire du DAO échantillonne un nombre aléatoire α dans l'ensemble $\mathbb{Z}_q$. L'échantillonnage aléatoire est important pour assurer la sécurité du système.

- Le gestionnaire du DAO calcule la clé publique *Schnorr* PK en effectuant l'opération $PK \leftarrow \alpha . G$, où α est la clé secrète choisie aléatoirement et G est le générateur du groupe cyclique fini $\mathbb{E}$.
- Le gestionnaire du DAO publie la clé publique PK sur le site du DAO. La clé publique est rendue accessible au public afin que les participants du DAO puissent interagir avec lui.
- Le gestionnaire du DAO conserve la clé secrète α en toute sécurité. La clé secrète doit rester confidentielle et ne doit pas être divulguée, car elle lui permet de prouver son identité lors de l'exécution de certaines opérations.

Il est important de noter que cette étape de création du DAO ne nécessite aucune activité sur la chaîne Ethereum, ce qui signifie qu'il n'y a pas de transactions à effectuer et aucun frais de gaz à payer. Cela permet une expérience utilisateur simplifiée et économique lors de la création d'un DAO sur la plateforme PDAO.

Étape 2 : Un membre envoie des fonds au DAO

Un client souhaite envoyer des ETH au DAO, disons un total de v ETH. Le client obtient la clé publique PK du DAO à partir du site web du DAO et effectue les opérations suivantes :

- Échantillonner un nombre aléatoire ρ dans $\mathbb{Z}q$.
- Calculer ce que nous appelons une feuille, qui est une paire :

$$L = (\rho . G, \rho . PK) \in \mathbb{E}^2 ;$$

- Appeler la fonction deposit() du contrat principal avec l'argument L, en envoyant v Ethers dans la transaction.
- Calculer la preuve de Merkle pour le prochain emplacement, soit la position suivante de l'arbre.La preuve de Merkle est simplement la liste des hachages depuis l'emplacement suivant jusqu'à la racine de l'arbre. Donc, cette preuve "π" contient “d” hachages.

- Le contrat utilise L et la preuve de Merkle stockée dans son état pour calculer la racine de Merkle mise à jour,Ensuite, le contrat vérifie que la preuve de Merkle fournie pour la position suivante est correcte et cohérente avec la racine de Merkle mise à jour.
- Le contrat principal accepte les fonds et enregistre la feuille étendue.

$$L' = (\rho \cdot G, \rho \cdot PK, v, n) ;$$

Dans la prochaine feuille non occupée, de gauche à droite, de l'arbre de Merkle T. Ici, “n” est le numéro de feuille désigné pour L' dans T. Plus précisément, le contrat fixe n ← next et insère L' dans la feuille numéro next. Le contrat incrémente la valeur de next de un et calcule la racine de Merkle de l'arbre T obtenu en ajoutant cette nouvelle feuille L' à l'arbre existant T à la position n. Si l'arbre a une profondeur d, le calcul de la racine de Merkle mise à jour peut être effectué en stockant uniquement d valeurs de hachage dans le stockage du contrat.

Comme la feuille L est publiée sur la chaîne et est visible publiquement par le monde entier, il est impératif que L ne révèle rien sur l'identité du DAO qui a reçu les fonds. En particulier, rien sur PK ne doit être révélé. Cela découle facilement de la difficulté de l'assomption de décision *Diffie-Hellman*[26] dans le groupe $\mathbb{E}$.

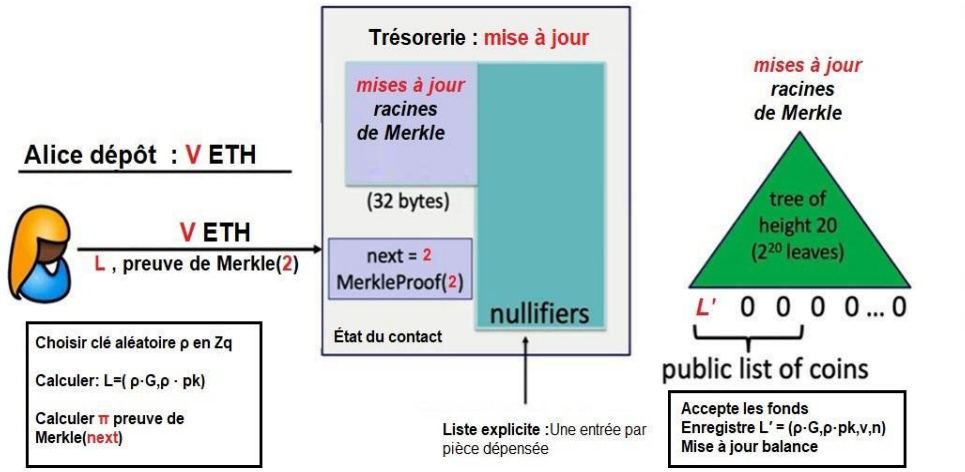

Figure 16 : l’état du contact trésorerie lors de la première transaction.

Étape 3 : Le gestionnaire du DAO retire des fonds

Lorsque le gestionnaire du DAO souhaite retirer w ETH du DAO, il doit le faire en prouvant que le DAO était le destinataire prévu d'au moins cette quantité d'ETH. De plus, ces fonds ne doivent être retirés qu'une seule fois. Tout cela doit être fait sans révéler le solde du DAO au contrat.

Nous y parviendrons en utilisant un SNARK.

Tout d'abord, comment le gestionnaire du DAO connaît-il même le solde des fonds envoyés au DAO ? Pour ce faire, le gestionnaire du DAO surveillera tous les dépôts envoyés au contrat principal. Pour chaque feuille observée

L' = (P, Q, v, n),

le gestionnaire du DAO utilise sa clé secrète α pour vérifier si :

α . P = Q.

Si tel est le cas, le gestionnaire du DAO apprend que ce dépôt est destiné à son DAO et enregistre que v ETH ont été ajoutés au solde du DAO. Sinon, ce dépôt est destiné à un autre DAO.

3.4.1 Le protocole de retrait

Supposons maintenant que le gestionnaire du DAO souhaite retirer w ETH du DAO. Nous exigeons qu'il existe un sous-ensemble de feuilles $\{L'_i=(P_i, Q_i, v_i, n_i)\}^{\ell}_{i=1}$ qui appartiennent au DAO tel que $v_1+ \ldots+v_\ell=w$.

Un protocole de retrait simple consiste à ce que le gestionnaire du DAO appelle la fonction withdraw() sur le contrat principal en lui donnant l'argument :

$(n1,\ldots,n\ell,\pi)$, où

π : représente une preuve SNARK que le gestionnaire DAO possède, avec un témoin :

$(\alpha , P1,Q1,v1, \ldots , P\ell,Q\ell,v\ell)$

Il est impératif que les conditions suivantes doivent être respectées :

1. Pour i=1,…,ℓ la feuille (Pi,Qi,vi,ni) est dans l'arbre de Merkle T ,
2. Pour i=1,…,ℓ on a Qi=α·Pi , et
3. v1+⋯+vℓ=w.

Le contrat principal vérifie la preuve SNARK π et vérifie que les feuilles n1,…,nℓ∈[2^d] n'ont pas encore été dépensées. Si tel est le cas, le contrat principal envoie w ETH au gestionnaire DAO, à l'origine de la demande de retrait. Il ajoute ensuite n1,…,nℓ à sa liste de feuilles épuisées nullifier afin que ces feuilles ne puissent plus être dépensées.

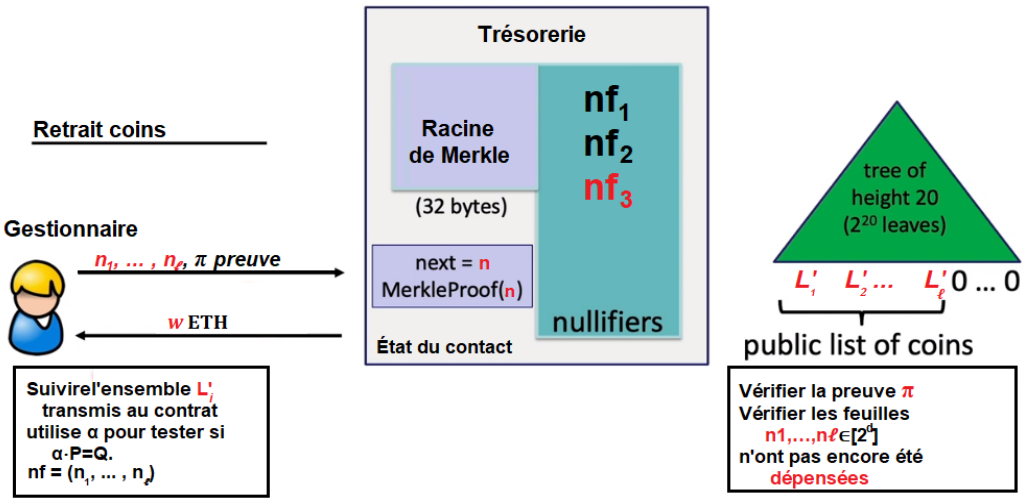

Figure 17 : Processus de retrait par le gestionnaire.

Puisqu'un SNARK nécessite une entrée de taille fixe, nous pouvons définir ℓ=100 , afin que le gestionnaire DAO puisse retirer un lot d'une centaine de feuilles à la fois.

Les parties qui suivent, vont être consacrées à exposer le contexte de notre solution ainsi que les différents objectifs que nous avons tracé à achever toute au long notre analyse du projet, les principaux services que la solution offre pour ces utilisateurs, afin de répondre au problème de trésorerie public déjà examiné auparavant.

3.5 Le But du produit

Pour que les DAO soient largement adoptés, notamment par les entreprises, il est essentiel de préserver la confidentialité des actifs détenus par leurs trésoreries. En particulier, limiter l'accès aux informations lié à ces fonds à des membres particuliers appelés les gestionnaires du DAO où seuls eux devraient connaître le montant total des actifs gérés par celui-ci. Il existe de nombreuses situations où une trésorerie privée est nécessaire :

- Un DAO d'investissement peut souhaiter que la taille de son fonds reste confidentielle.
- Une entreprise peut vouloir préserver la confidentialité de ses chiffres de ventes actuels.
- Un fonds participant à des enchères a besoin que sa trésorerie reste confidentielle.

Ce travail représente une modeste tentative de résoudre les problèmes liés à l'exposition des informations sensible sur les actifs introduite par le concept de trésorerie public en proposant un mixeur de fond sous forme d'un trésorerie collectif où plusieurs DAO peuvent mélanger les fonds reçu de leurs contributeurs au sein d'un seul contrat de trésorerie dans le but d'introduire un anonymat sur le montant des fonds reçus par ces DAO.

3.6 La portée du produit

Le protocole utilisé dans notre système permet de garantir la confidentialité des contributions, la protection des fonds et la gestion privé des actifs au sein d'une organisation décentralisée autonome. Ce système repose sur l'utilisation de technologies telles que le mixing, les SNARK et les arbres de Merkle pour créer un environnement sécurisé et confidentiel.

L'objectif principal du ce système est de permettre aux utilisateurs de créer et de gérer des trésoreries DAOs, tout en préservant la confidentialité de leurs informations sensibles. Il vise à répondre aux préoccupations liées à la divulgation des données qui peuvent nuire au bon fonctionnement de ces organisations, à la protection des fonds et à la

transparence sélective au sein des DAOs. Voici quelque objectifs clés du protocole utilisé dans notre système :

1. **Confidentialité du solde** : Lorsqu'un utilisateur envoie des fonds à un DAO géré par ce contrat, un observateur ne peut pas déterminer à quel DAO spécifique ces fonds sont destinés. L'observateur peut seulement connaître le montant total soumis à tous les DAO sur la plate-forme en consultant le solde du contrat principal.
2. **Gestionnaire de DAO** : Le gestionnaire de DAO possède la capacité de calculer le solde actuel des fonds envoyés à son DAO. Cette fonctionnalité permet au gestionnaire de prouver à une tierce partie que le solde du DAO est supérieur à un certain montant spécifié. Cependant, les détails spécifiques des contributions individuelles ou des soldes des contributeurs ne sont pas divulgués.
3. **Confidentialité des retraits** : Lorsque le gestionnaire de DAO effectue un retrait de fonds de son DAO, le montant retiré est public, ce qui signifie qu'il peut être observé par n'importe qui sur la blockchain. Cependant, le solde restant du DAO n'est pas divulgué. Cela signifie qu'un observateur ne peut pas déterminer le solde exact du DAO après le retrait.

Ces propriétés de confidentialité sont fondamentales pour préserver la vie privée des utilisateurs et protéger leurs informations financières sensibles lors de leurs interactions avec les DAOs sur la plateforme.

3.7 Description générale

3.7.1 Perspective

Le protocole utilisé dans notre système s’inscrit dans la famille des mélangeurs d'anonymat (mixers) tel que le protocole introduit par Tornado.cash Classic dans le réseau Ethereum, mais il présente une différence significative en termes de confidentialité. Contrairement à Tornado.cash, où l'expéditeur et le destinataire sont la même personne, notre protocole permet à n'importe qui d'envoyer des fonds au contrat principal, mais seul le gestionnaire du DAO peut les retirer. Ainsi, l'expéditeur et le destinataire sont distincts, où le gestionnaire du DAO agissant en tant que destinataire final des fonds.

3.7.2 Identification des acteurs du système

Un acteur représente une entité externe qui interagit directement avec le système. Dans notre cas, nous avons identifié deux acteurs principaux qui interagissent avec ce système. voici une description générale des deux types d'acteurs en interaction avec lui, et la figure 18 ci-après représente le diagramme de contexte statique du système à réaliser.

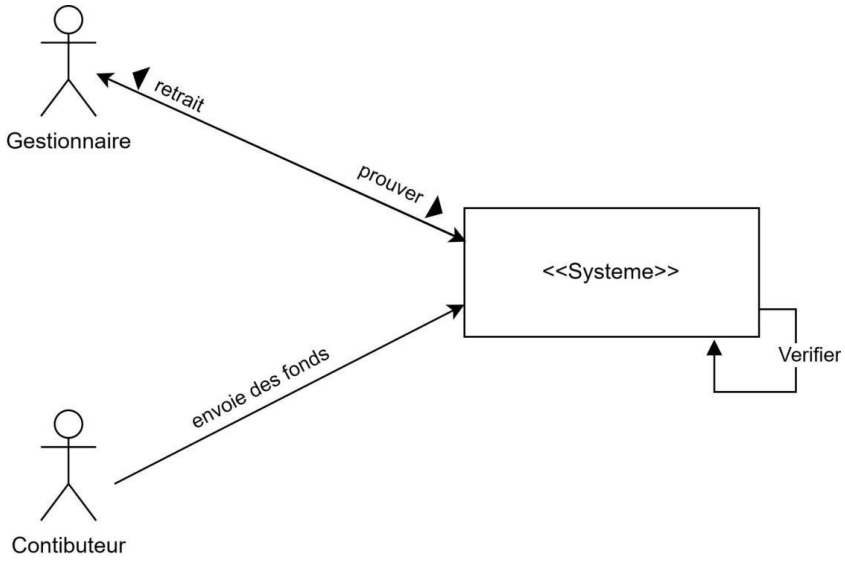

Figure 18 : Diagramme de contexte.

Pour chacun des acteurs cités au-dessus, notre système doit donc offrir un ensemble de fonctionnalités :

1. **Gestionnaire** :
    - Création du DAO (hors blockchain)
    - Surveillance des dépôts
    - Retraits de fonds

2. **Contributeur** :
    - Dépôts de fonds.

3.8 Fonctions du produit

Notre système offre trois principales fonctionnalités dont les organisations décentralisées autonomes ont besoin. Ces fonctionnalités sont vitales pour la gestion confidentielle et privée de leur trésorerie. Elles peuvent se résumer comme suit :

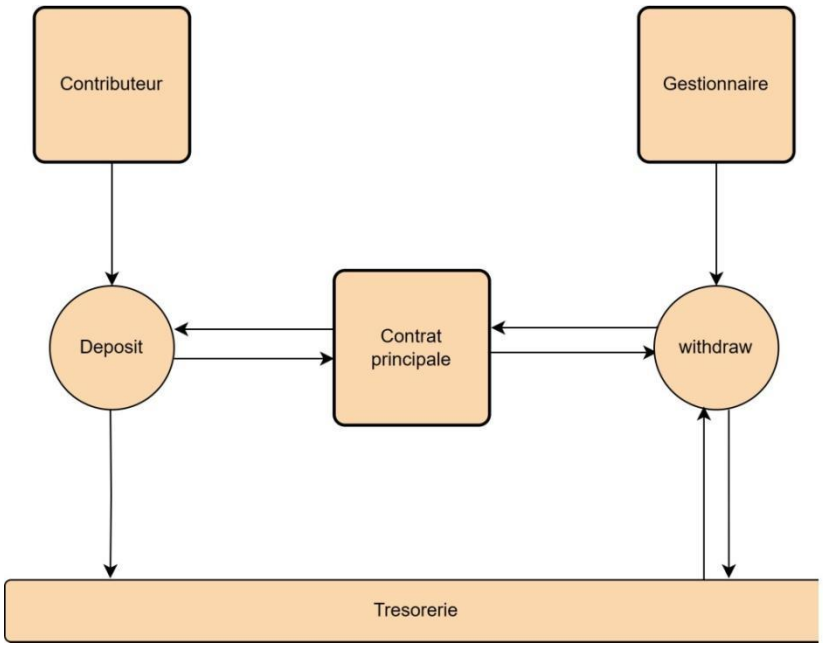

Figure 19 : Diagramme flux des données.

- **Création du DAO** : Le gestionnaire du DAO créé le DAO. Cette étape ne nécessite aucune transaction sur la blockchain.
- **Dépôt (Deposit) :** Utilisé par un utilisateur final qui souhaite contribuer des fonds au DAO.
- **Retrait (Withdrawal)** : Utilisé par le gestionnaire du DAO pour retirer des fonds du DAO.

3.8.1 Classes d'utilisateurs et caractéristiques

Notre produit est destiné à être utilisé principalement par les organisations autonomes décentralisés et toute structure souhaitant transitionner vers un système ouvert sur la blockchain gouverné par des smart contrat, tout en gardant leurs fonds et actif

confidentiel dans le but de mitiger les problèmes liés à la divulgation de ces informations financière sensibles et qui peuvent nuir au bon fonctionnement de ces organisations.

3.9 Environnement d’exploitation

Notre système utilise la technologie de la blockchain, en particulier Ethereum, pour enregistrer les transactions et assurer la décentralisation et l'immutabilité des opérations. Ethereum est une plateforme de blockchain qui permet l'exécution de contrats intelligents, des programmes informatiques autonomes qui s'exécutent automatiquement lors de l'exécution de conditions prédéfinies.

En utilisant Ethereum, le système peut créer des contrats intelligents qui gèrent les fonctionnalités clés de notre protocole, telles que les dépôts de fonds et les retraits. Les contrats intelligents sont écrits dans un langage de programmation spécifique appelé Solidity et sont exécutés sur la machine virtuelle Ethereum.

3.10Caractéristiques du système

Notre système comporte deux fonctionnalités principales :

- **Dépôt des fonds :**

Cette opération permet aux utilisateurs de poser des actifs sous forme de contribution dans le contrat principal en assignant cryptographiquement le DAO de choix, mais sans pourtant divulguer de quel DAO il s’agit.

- **Retirer des fonds :**

Cette opération permet à un type d’utilisateur bien spécifique de pouvoir retirer des actifs du contrat principal en prouvant qu’il s’agit bien d’un gestionnaire d’un DAO déjà enregistré dans le contrat sans divulguer toute même de quel DAO il s’agit. Ces gestionnaires sont seulement autorisés à retirer des fonds relatifs à leurs DAO.

3.11Diagrammes de cas d'utilisation global

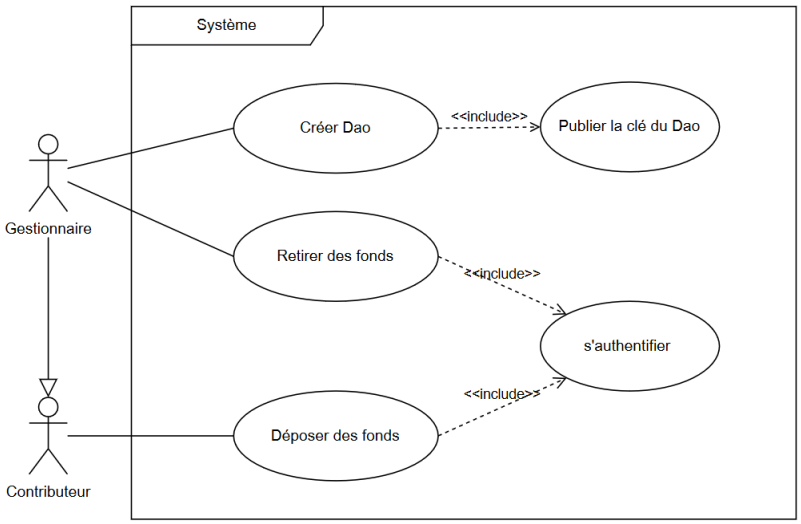

Figure 20 : Diagramme des cas d'utilisation.

3.12 Description des diagrammes de séquences

Ces diagrammes présentent l'enchaînement des opérations liés à un certain besoin fonctionnel que le système doit vérifier, ainsi que les différentes interactions entre les acteurs impliqués.

3.12.1 Diagramme de séquence du cas « S'authentifier »

Ce diagramme présente le processus d'authentification sur la platforme impliquant la connexion du portefeuille (wallet) utilisateurs du système la figure ci-dessous montre comment cela se déroule.

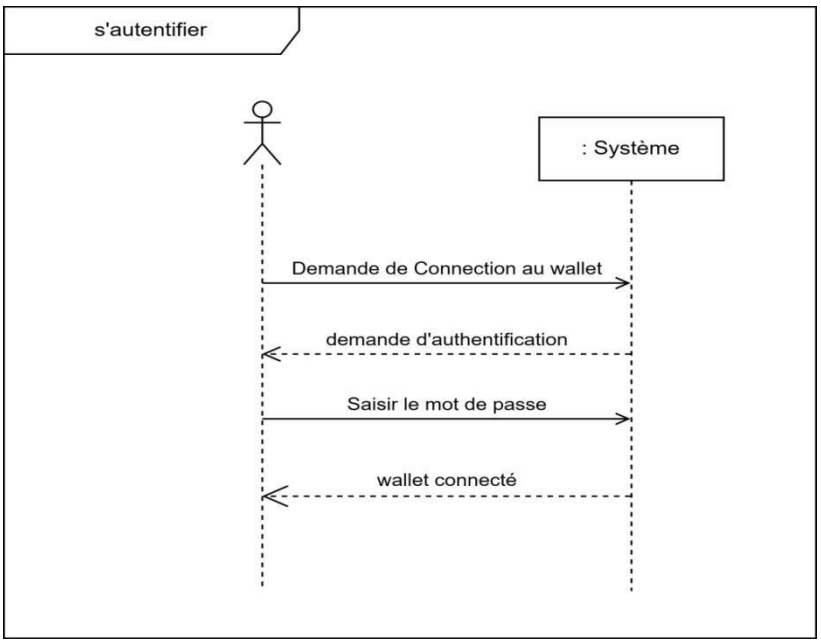

Figure 21 : Diagramme de séquence du cas « S'authentifier ».

3.12.2 Diagramme de séquence du cas Publier la clé du Dao

Ce diagramme présente les différentes étapes qui permettent au gestionnaire de créer son DAO.

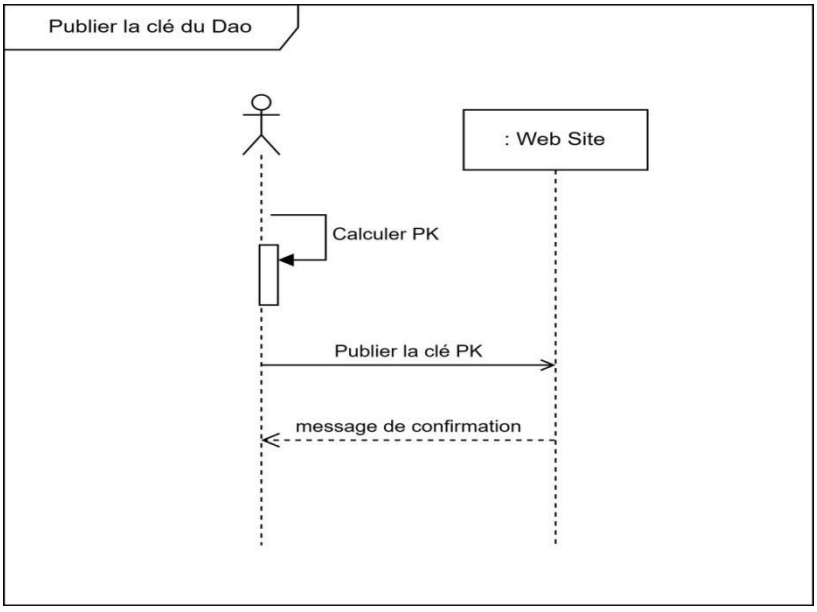

Figure 22 : Diagramme de séquence du cas Publier la clé du Dao.

3.12.3 Diagramme de séquence du cas Déposer des fonds

Le diagramme présenté ci-dessous représente le processus lorsqu'un contributeur choisit une DAO et souhaite participer à cette action. La figure 23ci-dessus détaille les étapes de dépôt de fonds dans cette DAO.

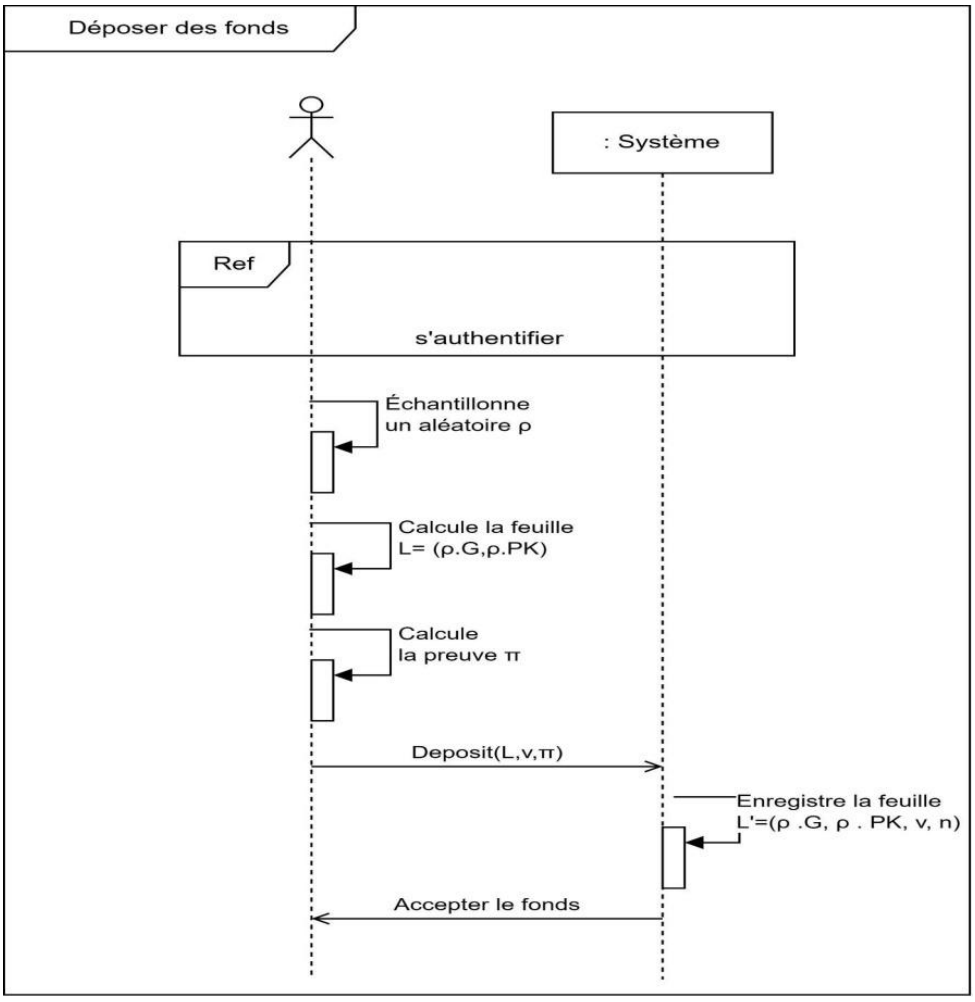

Figure 23 : Diagramme de séquence du cas Déposer des fonds.

3.12.4 Diagramme de séquence du cas Retirer des fonds

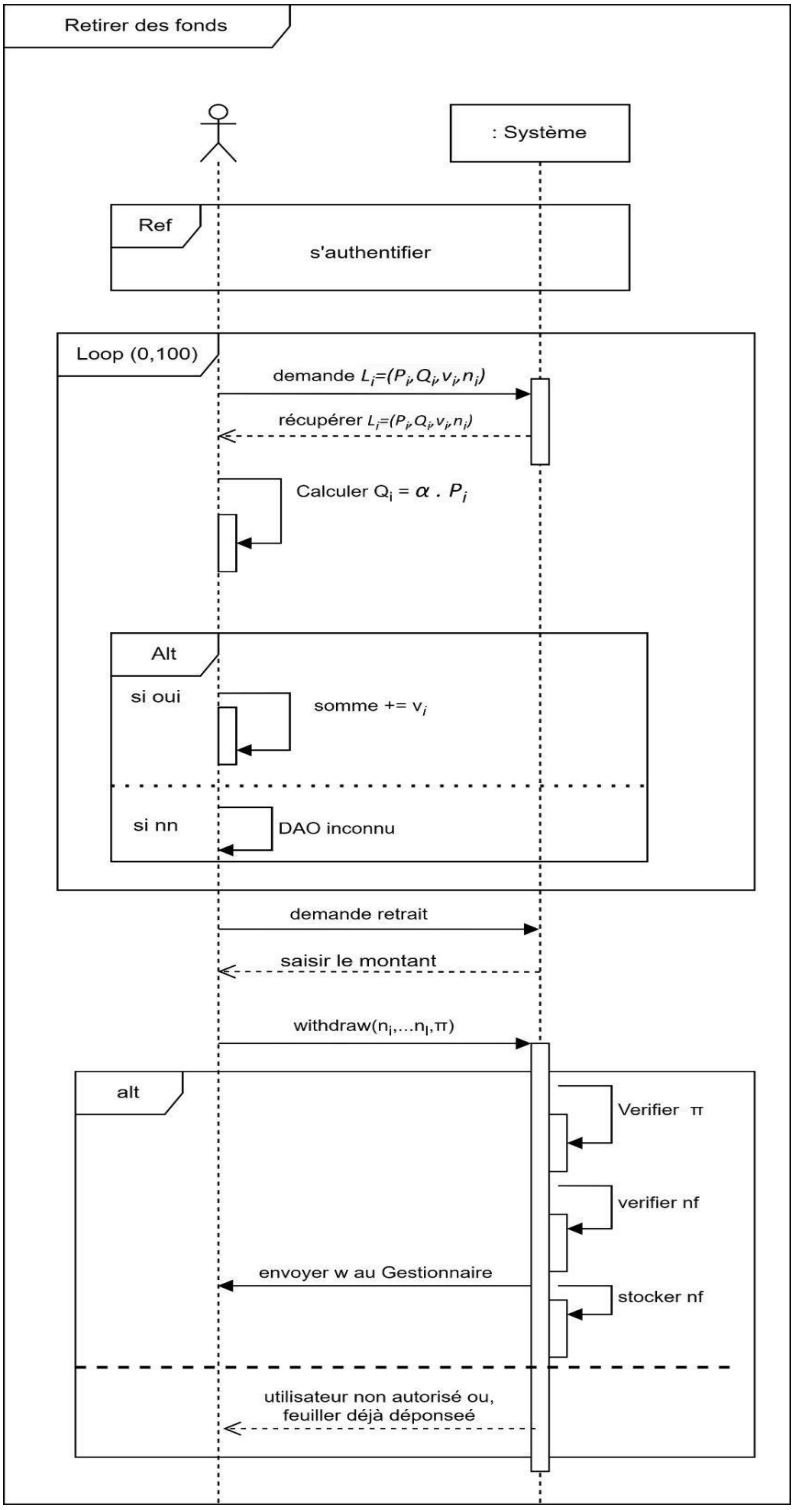

Figure 24 : Diagramme de séquence du cas Retirer des fonds.

3.13 Diagrammes de classe

Ce diagramme représente les entités principales impliquées dans la modélisation du système et sa structure statique à développer. Chaque entité se décrit par les données et les traitements dont elle est responsable.

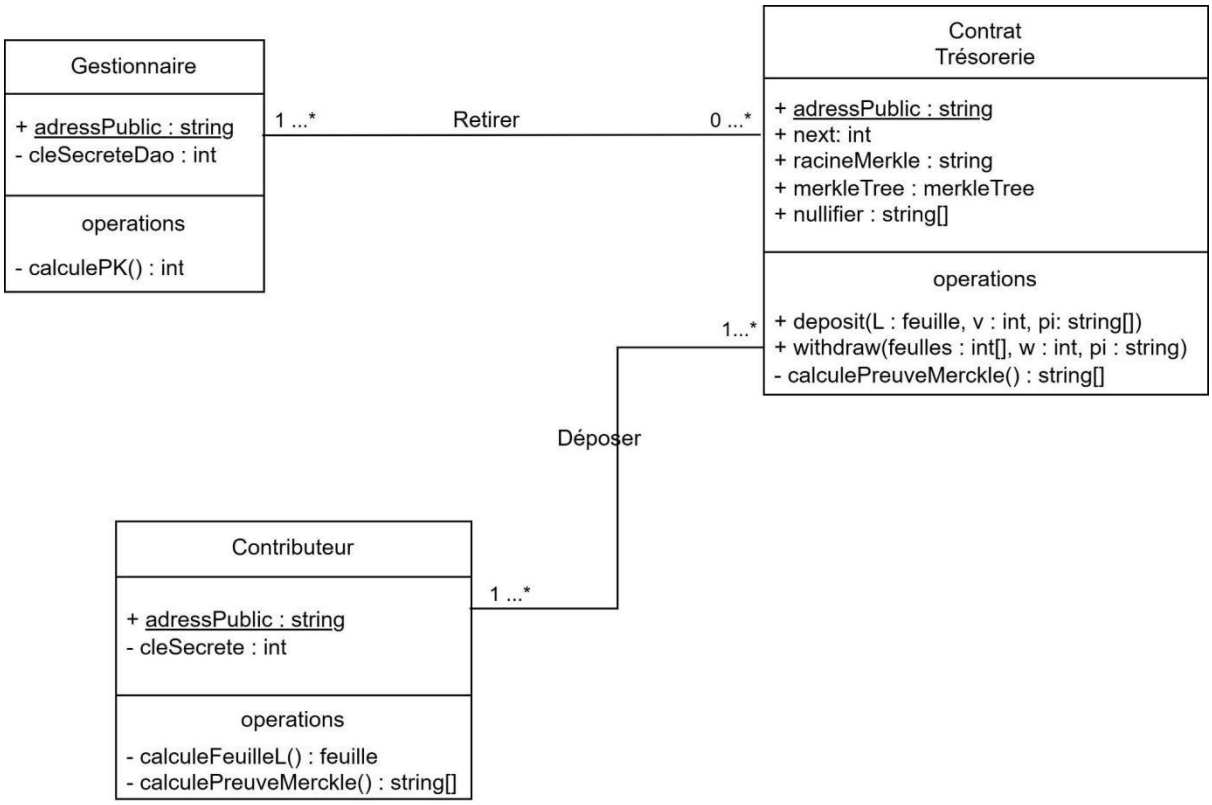

Figure 25 : Diagramme de classe.

Gestionnaire : Cette classe représente le gestionnaire DAO qui est responsable de la création et de la gestion du DAO. Elle contient des méthodes telles que calculePK() pour calculer la clé publique associée au DAO.

Contributeur : Cette classe représente l'utilisateur final qui souhaite contribuer avec des fonds au DAO. Elle contient la méthode calculeFeuilleL() qui effectue le calcul de la feuille représentant cryptographiquement pour quel dao ces fonds seront enregistrées, et la méthode calculePreuveMerke() qui donne la série de hash allant de la racine à la feuille finale ou le prochain noeud sera enregistré.

ContratTrésorerie : Cette classe représente la contract intelligente qui est responsable de vérifier les preuves fournies par le gestionnaire DAO lors d'un retrait de fonds. Elle contient également la méthode calculePreuveMerke() qui recalcule la preuve fournie par le contributeur. Ainsi que les deux principales méthodes deposit() et withdraw() qui permettent respectivement de recevoir des contributions et retirer des fonds du contrat.

3.14 Discussion de la solution proposée

Le protocole présenté offre plusieurs avantages en termes de confidentialité des ressources et de protection des informations sensibles aux siens des organisations décentralisées. Cependant, il comporte également certains inconvénients. Tout d'abord, le protocole ne prend pas en charge le remboursement direct des fonds aux contributeurs. Cela peut poser un problème si un contributeur souhaite récupérer tout ou une partie de ses fonds après les avoir initialement investis. De plus, le rôle du gestionnaire DAO peut être complexe lorsqu'il souhaite retirer un montant spécifique. Dans de tels cas, le gestionnaire n'a pas de visibilité sur les fonds restants après le retrait, ce qui peut compliquer la gestion des ressources. De même, si un contributeur souhaite rembourser le reste de ses fonds après un retrait,il peut être difficile de le réaliser efficacement.

Ces inconvénients soulignent la nécessité de considérer attentivement les limitations du protocole et de trouver des solutions pour améliorer la flexibilité et la facilité d'utilisation dans les scénarios de remboursement et de retrait des fonds.

3.15 Conclusion

Au cours de ce chapitre, nous avons approfondi notre discussion sur un protocole similaire à celui utilisé par le fameux mélangeur d'anonymat appelé Tornado cash du réseau Ethereum. Nous avons exposé les problèmes liés à la transparence financière des DAO dans un trésor public où toutes les transactions sont visibles, ainsi que les défis spécifiques liés à la participation des DAO à des enchères scellées, en prenant l'exemple de ConstitutionDAO.

Par la suite, nous avons défini les objectifs et analysé en détail le cycle de vie d'une trésorerie DAO gérée sur la plateforme. Ce cycle comprend trois étapes clés, la création du DAO, le dépôt de fonds et le retrait de ces fonds. Nous avons fourni une description détaillée de chacune de ces étapes, mettant en évidence les mesures prises pour garantir la confidentialité, protéger les fonds et permettre une gestion privée des actifs par le gestionnaire.

Enfin, nous avons exploré l'utilisation des preuves de non-connaissance (zero-knowledge proofs) pour maintenir la confidentialité des ressources financières. Ces techniques permettent de prouver la validité d'une information sans révéler les détails spécifiques, offrant ainsi une couche supplémentaire de protection des données financières au sein des DAO.

Chapitre 4

Implémentation

4.1 Introduction

Dans ce chapitre, nous nous consacrerons à la réalisation et à la mise en œuvre de notre application de gestion de gestion du trésorier des DAO, qui souhaite utiliser notre système pour gérer leurs fonds. Nous présenterons les outils de développement que nous avons adoptés, à savoir la Blockchain Ganache, le langage de programmation Solidity pour les smart contracts côté backend, le framework React et JavaScript pour le développement du frontend.

4.2 Outils de développement

4.2.1 Présentation du langage Solidity

La Solidité est un langage de programmation utilisé pour écrire des contrats intelligents sur la blockchain Ethereum. C'est un langage de programmation haut-niveau spécialement conçu pour la création de contrats intelligents et la mise en œuvre de logiques complexes dans un environnement décentralisé. La Solidité permet aux développeurs de spécifier les règles et les conditions d'exécution d'un contrat intelligent, définissant ainsi le comportement attendu et les interactions avec d'autres contrats et utilisateurs sur la blockchain.

Figure 26 : Logo Solidity.

4.2.2 Présentation de Ganache

La blockchain Ganache est une blockchain de développement utilisée principalement dans le domaine de la blockchain Ethereum. Elle est souvent utilisée pour le développement et les tests de contrats intelligents.

Ganache fournit un environnement de développement local qui simule une blockchain Ethereum complète. Il permet aux développeurs de créer et de tester des applications blockchain sans avoir besoin de se connecter à un réseau Ethereum en direct.

Figure 27 : Logo Ganache.

4.2.3 JavaScript

JavaScript est un langage informatique utilisé sur les pages web. Ce langage a la particularité de s'activer sur le poste client, en d'autres mots c'est

votre ordinateur qui va recevoir le code et qui devra l'exécuter. C'est en opposition à d'autres langages qui sont activés côté serveur.

Figure 28 : Logo JavaScript.

4.2.4 Bibliothèque web3.js

Pour créer une application web qui peut interagir avec la Blockchain Ethereum, nous avons besoin d'utiliser une bibliothèque JavaScript appelée web3.js. Cette bibliothèque nous permettront d'entrer l'adresse d'un Smart Contract et d'appeler les fonctions qu'il contient, en fournissant les éventuels paramètres requis.

Figure 29 : Logo Web3.js.

4.3 React

React est une bibliothèque JavaScript open-source utilisée pour construire des interfaces utilisateur interactives et réactives. Développée par Facebook, React permet de créer des composants réutilisables qui encapsulent la logique et l'affichage d'une partie de l'interface utilisateur.

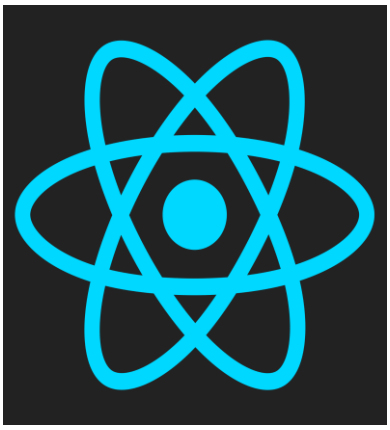

Figure 30 : Logo React.

4.4 Environnement de développement

4.4.1 Présentation Visual Studio Code

Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur votre bureau et est disponible pour Windows, macOS et Linux. Il dispose d'une prise en charge intégrée pour JavaScript, TypeScript et Node.js, ainsi que d'un écosystème riche en extensions pour d'autres langages et environnements d'exécution (tels que C++, C#, Java, Python, PHP, Go, .NET). Débutez votre aventure avec VS Code grâce à ces vidéos d'introduction.

Figure 30 : Logo Visual Studio Code.

4.4.2 MetaMask

MetaMask est une extension de navigateur qui permet aux utilisateurs d'interagir avec des applications décentralisées (DApps) construites sur la blockchain Ethereum. Elle agit comme un portefeuille numérique, permettant aux utilisateurs de gérer leurs comptes Ethereum et d'effectuer des transactions en utilisant des contrats intelligents.

Figure 31 : Logo MetaMask.

4.4.3 Remix-Ethereum

Remix-Ethereum est un environnement de développement intégré (IDE) en ligne pour la programmation et le déploiement de contrats intelligents sur la blockchain Ethereum. Il fournit une interface conviviale et puissante pour écrire, compiler, déployer et tester des contrats intelligents en langages tels que Solidity.

Figure 32 : Logo Remix-Ethereum.

4.5 Organigrammes de l'application

Dans cette partie, nous allons fournir un aperçu global des fonctionnalités de l'application en utilisant une série de captures d'écran. Ces captures d'écran nous permettront de visualiser les différentes fonctionnalités de l'application, Lorsqu'on lance l'exécution notre Dapp sur le serveur de développement, on obtient l'affichage suivant :

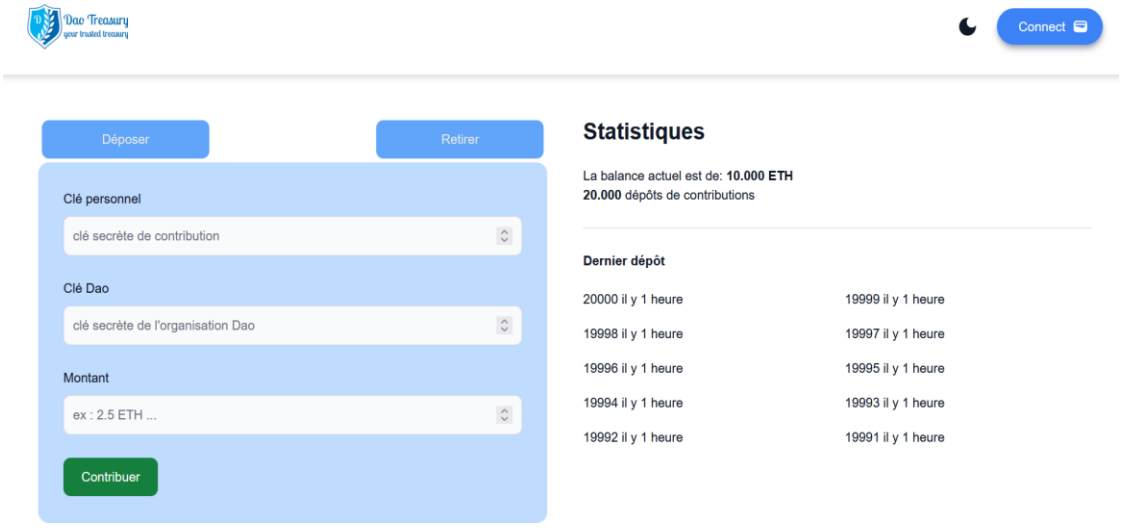

Figure 33 : Interface principale de l'application.

4.5.1 Interface MetaMask

Le contributeur doit se connecter à la Blockchain en utilisant l'extension « MetaMask », ce qui lui permet d'utiliser son propre portefeuille Ethereum pour signer ses transactions.

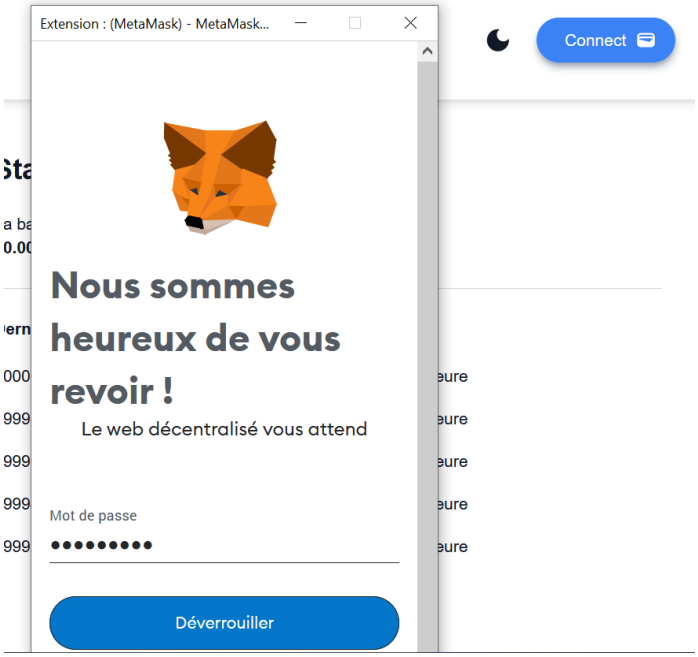

Figure 34 : Interface de connexion MetaMask.

4.5.2 Interface contributeur

Cette figure représente la fenêtre d'exécution de l'application permettant aux utilisateurs de déposer des fonds dans la balance du contrat principal. L'utilisateur doit fournir sa clé publique en tant que contributeur, la clé du DAO (Organisation Autonome Décentralisée) où il souhaite effectuer sa contribution, ainsi que le montant qu'il souhaite déposer. Ces informations sont nécessaires pour que l'utilisateur puisse participer et contribuer au contrat principal.

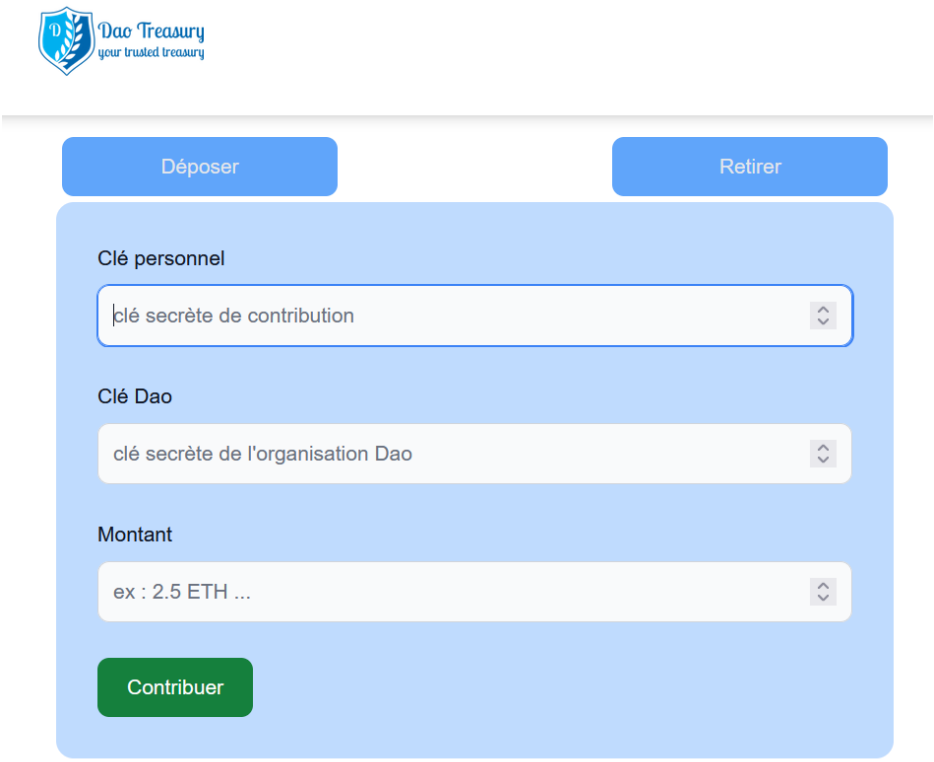

Figure 35 : Interface contributeur.

4.5.3 Interface gestionnaire

Cette figure représente la fenêtre du gestionnaire lorsqu'il souhaite retirer des fonds. Pour pouvoir effectuer cette opération, le gestionnaire doit fournir sa clé secrète afin de prouver son identité en tant que gestionnaire de ce DAO (Decentralized Autonomous Organization). En fournissant sa clé secrète, le gestionnaire confirme qu'il a l'autorité nécessaire pour effectuer des retraits de fonds. De plus, le gestionnaire doit également spécifier le montant qu'il souhaite retirer de la balance du DAO. En renseignant ces informations, le gestionnaire peut procéder au retrait des fonds désirés.

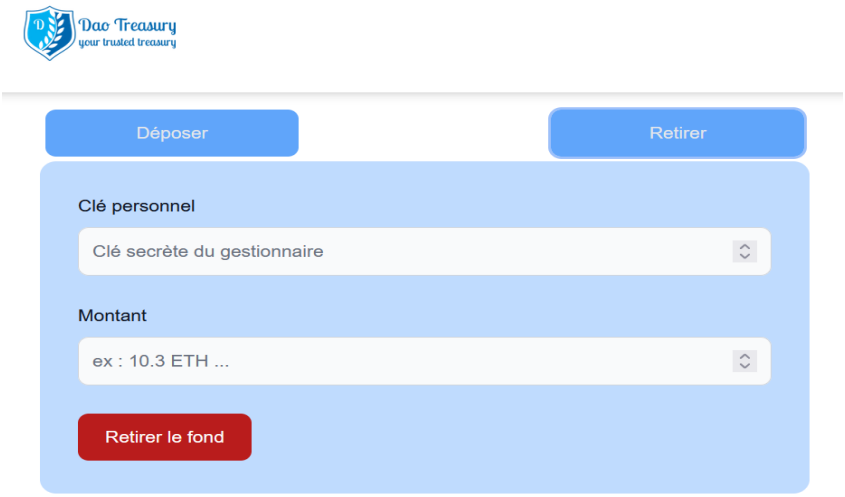

Figure 36 : Interface gestionnaire souhaite retirer.

4.6 SmartContrast

- **Fonction : Deposit()**

Cette figure représente le déploiement d'un smart contrat dans l'éditeur Remix, comme mentionné précédemment. Elle illustre également l'exécution de la fonction "deposit" du smart contrat, qui enregistre l'état de la balance après un dépôt. Les informations chiffrées fournies par le contributeur et le montant déposé sont également affichées, avec la validation du contrat situé en dessous.

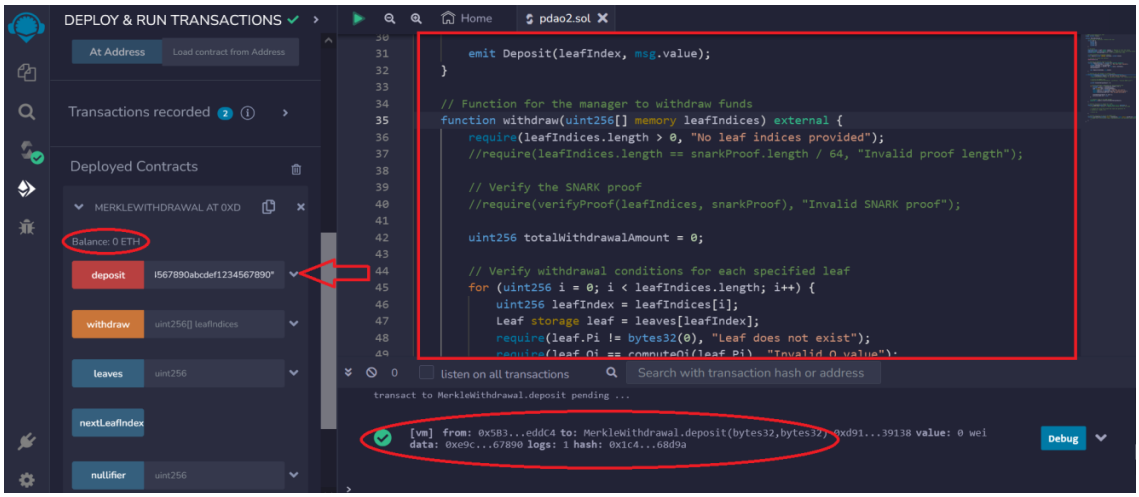

Figure 37 : Déploiement de contrat « fonction deposit ».

- **Fonction : Withdraw()**

Cette figure illustre également l'exécution de la fonction "withdraw" dans laquelle le gestionnaire peut retirer une liste de feuilles (données) qui ont été déposées par les contributeurs, ainsi que leurs valeurs. De plus, l'état des feuilles est affiché, incluant le hachage des contributeurs, ce qui permet au gestionnaire de connaître les feuilles qui appartiennent à son DAO (Decentralized Autonomous Organization).

La fonction "withdraw" permet au gestionnaire d'effectuer des retraits spécifiques dans le contrat en sélectionnant des feuilles spécifiques. Les feuilles sont des enregistrements de données déposés par les contributeurs. Lorsqu'un retrait est effectué, les feuilles correspondantes sont retirées de la liste

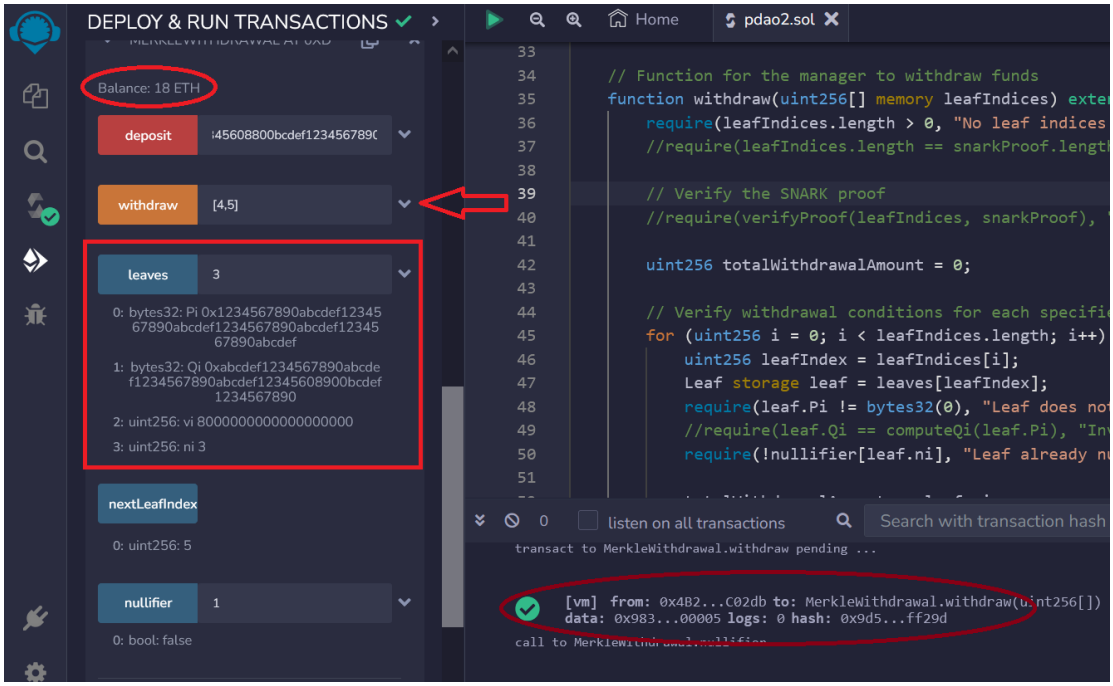

Figure 38 : Déploiement de contrat « fonction "withdraw" ».

4.7 Conclusion

Dans cette dernière partie, nous avons présenté la côté réalisation et implémentation de notre projet en spécifiant nos choix des outils, des langages et d'environnement de développement que nous avons adopté pour développer notre application.

Nous avons proposé notre solution qui représente une alternative du trésorerie publique, assurant un niveau de confidentialité par technique de mixing et zk-SNARK inspiré de tornado cash. Notre solution consiste en un *Smart Contract* en Backend et une application *React* Cliente en Frontend.

Notre solution prouve qu'un niveau de confidentialité est toujours atteignable même sur une blockchain publique.

Conclusion Général

Pour conclure, la technologie blockchain reste un domaine très jeune… qui a néanmoins ouvert la porte à d'autres champs de recherche et d'exploitation des techniques cryptographiques qui s'est limité auparavant à l'assurance de la sécurité des échanges sur le grand réseau web. Ces idées ont donné naissance à plusieurs opportunités aux yeux de différents acteurs du domaine financier et recherche technologique, sur lequel on a vu apparaître le concept des organisations autonomes décentralisées (DAO). En effet, ces entités décentralisées permettent aux organisations de ne plus dépendre des institutions traditionnelles ou un corps central pour orchestrer la coordination des membres et renforcer les règles de gouvernance. Ces règles de gestion sont transcrites ouvertement dans la même blockchain qui représente la DAO elle-même, pour tout le monde a consulté, grâce à l'utilisation des contrats intelligents et les langages de programmation tel que solidity. Cette transparence, même si elle représente un point fort hérité de l'utilisation de la technologie blockchain et appuyé par les créateurs de ces organismes, à entraîner ces derniers dans des problèmes de sécurité et de confidentialité très sérieux dans lequel des mesures urgentes ont été prises pour mitiger leurs effets secondaires.

Dans l'ensemble, ces chapitres ont permis d'approfondir notre connaissance des DAO, en mettant en évidence leur structure, leur fonctionnement, leurs défis et les solutions de confidentialité disponibles. Les DAO jouent un rôle croissant dans la gouvernance et l'organisation décentralisées, et leur compréhension est essentielle pour saisir les implications de la technologie blockchain dans différents domaines.

Bibliographie

[1] « Blockchain.com | Buy Bitcoin, Ethereum and more with trust ». https://www.blockchain.com (consulté le 16 février 2023).

[2] « Home | ethereum.org ». https://ethereum.org/en/ (consulté le 16 février 2023).

[3] « Bitcoin - Open source P2P money ». https://bitcoin.org/en/ (consulté le 16 février 2023).

[4] S. Riva, « Decentralized Autonomous Organizations (DAOs) in the Swiss Legal Order », in *Yearbook of Private International Law Vol. XXI - 2019/2020*, A. Bonomi et G. P. Romano, Éd. Verlag Dr. Otto Schmidt, 2020, p. 601-638. doi: 10.9785/9783504386962-028.

[5] Colin, Jean Noël « Du Bitcoin aux DAO ». IEEE Commun. Surv. Tutor., vol. 20, no 3, p. 2543-2585, 2018, doi: 10.1109/COMST.2018.2818623.

[6] « GODEBARGE ROSSAT Blockchain-Version-Finale | PDF | Public-key cryptography | Bitcoin », *Scribd*. https://fr.scribd.com/document/451953499/GODEBARGE-ROSSAT-Blockchain-version-finale (consulté le 20 février 2023).

[7] A. Wright, « THE RISE OF DECENTRALIZED AUTONOMOUS ORGANIZATIONS: OPPORTUNITIES AND CHALLENGES ».

[8] « Provable Documentation ». https://docs.provable.xyz/#home (consulté le 16 février 2023).

[9] « TLSNotary ». http://tlsnotary.org (consulté le 16 février 2023).

[10] M. U. Hassan, M. H. Rehmani, et J. Chen, « Differential Privacy in Blockchain Technology: A Futuristic Approach », 2019, doi: 10.48550/ARXIV.1910.04316.

[11] M. C. Kus Khalilov et A. Levi, « A Survey on Anonymity and Privacy in Bitcoin-Like Digital Cash Systems », *IEEE Commun. Surv. Tutor.*, vol. 20, n° 3, p. 2543-2585, 2018, doi: 10.1109/COMST.2018.2818623.

[12] S. Hassan et P. De Filippi, « Decentralized Autonomous Organization », *Internet Policy Rev.*, vol. 10, n° 2, avr. 2021, doi: 10.14763/2021.2.1556.

[13] S. Riva, « Decentralized Autonomous Organizations (DAOs) as Subjects of Law – the Recognition of DAOs in the Swiss Legal Order ». Rochester, NY, 10 octobre 2019.

Consulté le: 20 février 2023. [En ligne]. Disponible sur: https://papers.ssrn.com/abstract=3515229

[14] Galia Kondova et Renato Barba, « Governance of Decentralized Autonomous Organizations », *J. Mod. Account. Audit.*, vol. 15, nº 8, août 2019, doi: 10.17265/1548-6583/2019.08.003.

[15] « Qu'est-ce que l'ether (ETH) ? », *ethereum.org*. https://ethereum.org (consulté le 21 février 2023).

[16] I. Bashir, *Mastering Blockchain*. Packt Publishing Ltd, 2017.

[17] A. Narayanan, J. Bonneau, E. Felten, A. Miller, et S. Goldfeder, « Bitcoin and Cryptocurrency Technologies ».

[18] S. Wang, W. Ding, J. Li, Y. Yuan, L. Ouyang, et F.-Y. Wang, « Decentralized Autonomous Organizations: Concept, Model, and Applications », *IEEE Trans. Comput. Soc. Syst.*, vol. 6, nº 5, p. 870-878, oct. 2019, doi: 10.1109/TCSS.2019.2938190.

[19] Y. Faqir-Rhazoui, J. Arroyo, et S. Hassan, « A comparative analysis of the platforms for decentralized autonomous organizations in the Ethereum blockchain », *J. Internet Serv. Appl.*, vol. 12, nº 1, p. 9, déc. 2021, doi: 10.1186/s13174-021-00139-6.

[20] « WEF_Decentralized_Autonomous_Organizations_Beyond_the_Hype_2022.pdf ».

[21] D. Boneh, « How to build a private DAO on Ethereum », *Medium*, 1 mai 2022. https://medium.com/@boneh/how-to-build-a-private-dao-on-ethereum-59e3afdf3abd (consulté le 16 février 2023).

[22] G. Zyskind, O. Nathan, et A. Pentland, « Enigma: Decentralized Computation Platform with Guaranteed Privacy », 2015, doi: 10.48550/ARXIV.1506.03471.

[23] R. Xue et L. Liu, « Security and Privacy on Blockchain », *ACM Comput. Surv.*, vol. 52, nº 3, p. 1-34, mai 2020, doi: 10.1145/3316481.

[24] Aztec Yellow Paper », *HackMD*. https://hackmd.io/@aztec-network/ByzgNxBfd#2-Grumpkin---A-curve-on-top-of-BN-254-for-SNARK-efficient-group-operations (consulté le 19 mai 2023).

[25] EIP-197: Precompiled contracts for optimal ate pairing check on the elliptic curve alt_bn128 », *Ethereum Improvement Proposals*. https://eips.ethereum.org/EIPS/eip-197 (consulté le 05 juin 2023).

[26] D. Boneh, « The Decision Diffie-Hellman problem », in *Algorithmic Number Theory*, J. P. Buhler, Éd., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1998, p. 48-63. doi: 10.1007/BFb0054851.