**Transaction en crypto-monnaie vs standard**

**Illustration paiement standard**

1. Etapes
2. Transfert au vendeur
3. Parties concernées

* Acheteur
* Banque acheteur (A)
* Banque vendeur (V)
* Vendeur

1. Emission monnaie

* Banque centrale (base monétaire)
* Banques commerciales (masses monétaires M1, M2..) : multiplicateur monétaire, crédits - dépôts

1. Déroulement

*Etape 1*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Actifs** | **Passifs** | **Rôle** |
| Acheteur A | - 50 |  | Ordre de pmt à banque A |
| Banque A | - 50 | - 50  *+ 50* | Débit compte A  Virement à banque V (cptes BNS)  *Virement à banque correspondant, pour le compte de banque V\** |
| *Correspondant US (banque)\** | *+ 50*  *- 50* |  | *Si pmt via bque correspondante, p.ex. virement en USD* |
| Banque V | + 50 | *- 50*  + 50 | Virement de banque A (cptes BNS)  *Virement reçu sur banque correspondant\**  Crédit compte de V |
| Vendeur V | + 50 |  | - |

*\* Cas avec pmt via banque correspondante (ex. pmt US)*

*Hypothèse : correspondant US créancier envers banque A et banque B (avant et après*

*pmt)*

* Fondamentalement : échange de créances sur des banques

**Illustration paiement crypto-monnaie (*exemple du bitcoin*)**

1. Etapes
2. Achat cryptomonnaie via compte bancaire
3. Transfert au vendeur
4. Parties concernées

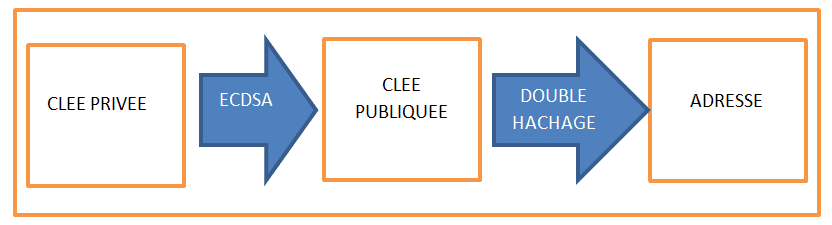
* Acheteur (également acheteur crypto)
* Banque acheteur (A)
* Banque vendeur crypto (V)
* Vendeur crypto
* *Intermédiaire crypto 1 (Wallet)*
* *Intermédiaire crypto 2 (Plateforme d’échange) – nécessaire si intervention autres monnaies*
* Blockchain
* Mineurs (= nœuds de la blockchain)
* Vendeur

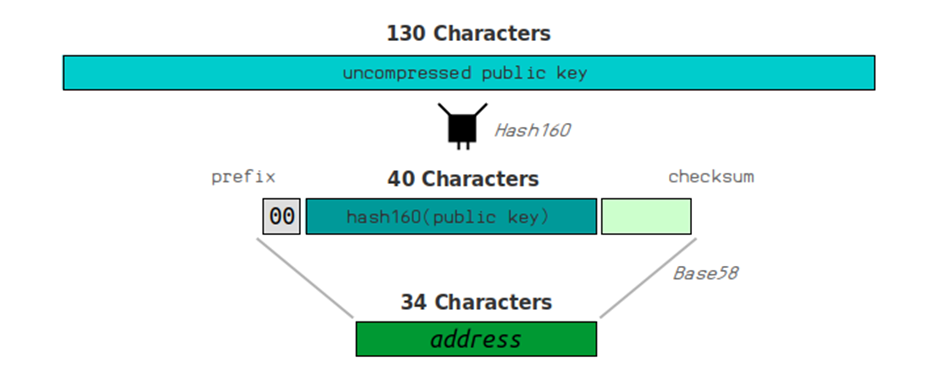
1. Emission crypto (bitcoin)

Par un algorithme, liée exclusivement au minage :

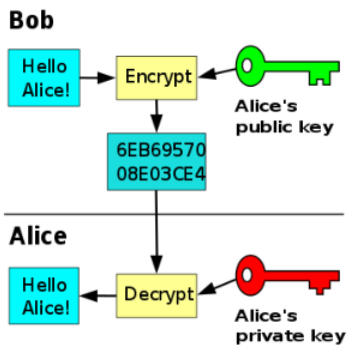
* création en paiement du travail des mineurs
* l’algorithme attribue des bitcoins de manière prédéterminée (taux d’émission)
* montant de bitcoins attribués par nouveau bloc miné divisé par deux tous les 210'000 blocs (env. 1 nouveau bloc apporté à la blockchain toutes les dix minutes)
* maximum de 21 millions de bitcoins attribuable (sera atteint en 2141)
* lors de la création d’un nouveau bloc, une transaction spéciale, le *coinbase*, est inclue dans le bloc. Elle servira à payer le mineur dont le travail a été sélectionné comme déterminant le contenu de la blockchain
* les mineurs sont également payés en frais de transactions (prélevés sur le montant de la transaction, cf ex. <https://www.blockchain.com/btc/address/1GzJPqJbXTJpDhN5EUUVaF2wqBsFdVGkrX>) . Les frais de transaction correspondent à la partie inutilisée des $crypto mises en input dans la transaction (voir transaction) et incitent les mineurs à intégrer la transaction dans leur(s) block(s). Une fois la totalité des bitcoins minée, ce ne seront plus que les frais de transactions qui fourniront une source de revenu aux mineurs.

1. Identification (système de clés asymétriques)
   * clé publique (*~ no de compte = adresse[[1]](#footnote-2)*) => permet de reconstituer historique de toutes les transactions, version transformée (ECDSA, fonction elliptique) de la clé privée (clé publique apparaît dans blockchain en version également hashée, l’adresse $crypto, cf blockchain-browser : <https://www.blockchain.com/explorer?view=btc_blocks> )
   * clé privée (*~ mot de passe = signature transaction*) => permet identification pour exécuter transaction, clé publique liée à clé privée par hashage (ce qui permet de contrôler que celui qui renseigne la clé privée est bien le «détenteur» de la clé publique)





* principe issu de la cryptographie asymétrique

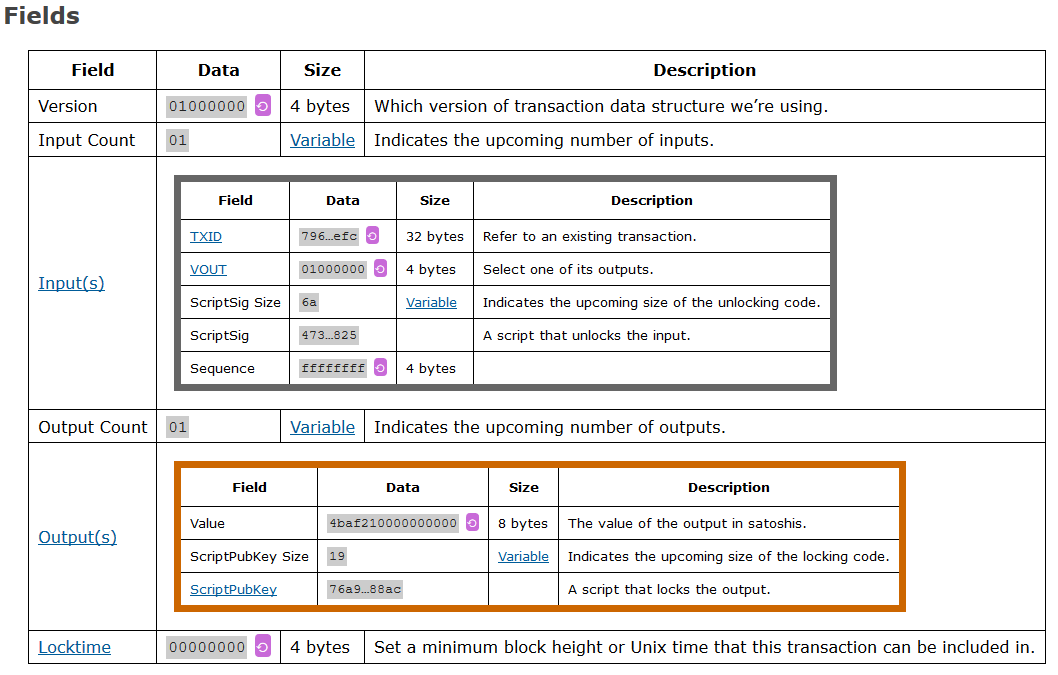


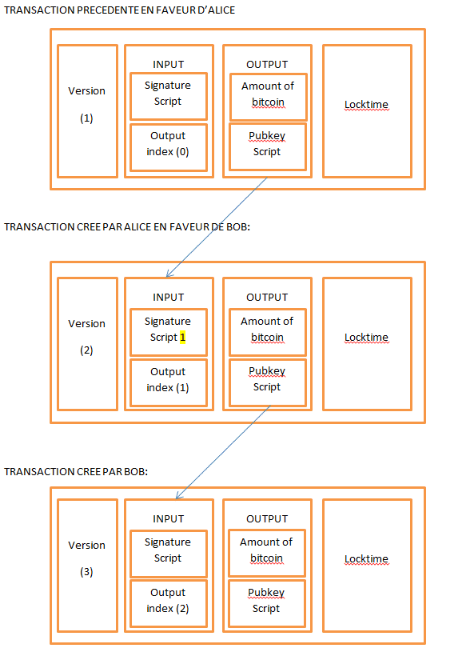
Le message est crypté avec la public key (communiquée par Alice), qui dérive de la private key par une fonction de hashage => pour décrypter, besoin de donner la valeur qui, après application de la fonction de hashage définie donne la public key => = private key !

1. Contenu transactions (bitcoin)

* non-cryptées, visibles par tous !
* transaction = échange d’accès à des $crypto, 2 parties : input (accès au $crypto) + output (transfert accès $crypto)
* lien entre input(s) et précédente(s) transaction(s) (réception accès $crypto) via PubkeyScript et SignatureScript

|  |  |
| --- | --- |
| **Elément** | **Description** |
| PubkeyHash | Version hashée de public key :  (1) Public Key→(2) SHA256→(3) RIPEMD160→ PublicKey Hash (160 bits) |
| Adresse $crypto | Version “raccourcie” de PubkeyHash :  PublicKey Hash →(4) Base58Check Encode with 0x00 version prefix = adresse $crypto |
| DigitalSignature | Version transformée de privateKey et du hash des données de la transaction (signature spécifique à la transaction). Permet de valider utilisation des $crypto sans communiquer privateKey en clair. |
| PubkeyScript | Locking script : script établi par «payeur», contient adresse$crypto receveur (dérivée de publicKey), doit être « unlocked » pour pouvoir utiliser les $crypto   * + comparaison DigitalSignature avec publicKey => même privateKey utilisée ?   + comparaison publicKey avec adresse$crypto => compatible ? |
| SignatureScript | Unlocking script : script établi par «récepteur», contient DigitalSignature et publicKey, permet de résoudre PubkeyScript (accès aux $crypto) |





1. Déroulement

*Etape 1 : achat cryptomonnaie*

Préliminaire : création *wallet* bitcoin = 3 fonctions

* conservation clés privées
* création clés publiques (et succédanés : adresse $crypto, scripts..)
* lien avec réseau (blockchain)

Types de *wallet[[2]](#footnote-3)* :

* online (*hot storage* : clés privées et publiques stockées sur le site d’un prestataire de service)
* installation logiciel sur PC (*cold storage*, sauvegarde clés privées peut être faite sur des clés USB sécurisées)
* papier (clés imprimées)
* Hierarchical Deterministic (HD) wallet : génération de clés publiques aléatoirement à partir de clé publique principale et seed fixé … pour plus tard…

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Actifs** | **Passifs** | **Rôle** |
| **Partie monnaie standard** | | | |
| Acheteur A | - 50 |  | Ordre de pmt à banque A |
| Banque A |  | - 50  + 50 | Débit compte A  Virement à banque V (via cptes correspondant US) |
| Banque V |  | - 50  + 50 | Virement de banque A (via cptes correspondant US)  Crédit compte de Vendeur crypto |
| Vendeur crypto | + 50 |  | Réception pmt |
| *Intermédiaire crypto 2 (plateforme)* | *-* | *-* | *Rôles ?*   * *Validation que pmt reçu pour envoyer transaction crypto ?* * *Dépôt temporaire monnaie en attendant validation crypto Dépôt temporaire monnaie en attendant validation crypto* * *Achat/vente en nom propre ?* * *Wallet peut être incorporée* |
| **Partie crypto** | | | |
| Vendeur crypto | - $Crypto |  | Transfert crypto |
| *Intermédiaire crypto 1 (wallet)* | *-* | *-* | *Rôles :*   * *Conservation clés* * *Accès blockchain, transmission infos transaction crypto-monnaie* * *Création adresse $crypto, PubkeyScript, SignatureScript* |
| *Intermédiaire crypto 2 (plateforme)* | *-* | *-* | *Rôles ?*   * *Validation que pmt reçu pour envoyer transaction crypto ?* * *Dépôt temporaire monnaie en attendant validation crypto* |
| *Blockchain* |  |  | *Rôles :*   * *Conservation historique transactions* * *Stocké, distribué chez nœuds/mineurs* |
| *Mineurs* |  |  | *Rôles :*   * *Participants à blockchain, stockage ledger distribué* * *Réception en temps réel des transactions, validation*   *1) validité transaction (...Script)*  *2) disponibilité des $Crypto (selon clé publique)*   * *Constitution de blocs, vérification bloc => intégration dans blockchain* |
| Acheteur A | + $Crypto |  | Réception crypto |
| Questions |  |  | Questions :   * **Quel lien entre pmt en USD et transfert crypto ? Intermédiaire pour synchroniser ?** |

*Etape 2*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Actifs** | **Passifs** | **Rôle** |
| Acheteur A | - $Crypto |  | Achat vs paiement en crypto |
| *Intermédiaire crypto 1 (wallet)* | *-* | *-* | *Rôles :*   * *Conservation clés* * *Accès blockchain, transmission infos transaction crypto-monnaie* * *Création adresse $crypto, PubkeyScript, SignatureScript* |
| *Blockchain* |  |  | *Rôles :*   * *Conservation historique transactions* * *Stocké, distribué chez nœuds/mineurs* |
| *Mineurs* |  |  | *Rôles :*   * *Participants à blockchain, stockage ledger distribué* * *Réception en temps réel des transactions, validation*   *1) validité transaction (….Script)*  *2) disponibilité des $Crypto (selon clé publique)*   * *Constitution de blocs, vérification bloc => intégration dans blockchain* |
| Vendeur V | + $Crypto |  | Vente vs paiement en crypto |

1. Qq notions

* *block* : regroupement de transactions destinées à être validées (chaque 10mn). Comme chaque mineur ne reçoit pas les transactions au même moment (situation géographique, vitesse de connexion...), les blocs générés ne sont pas tous identiques => règle de consensus (« course au *nonce »*) pour décider quel mineur aura le droit d'ajouter son propre bloc à la seule et unique blockchain[[3]](#footnote-4). Contenu du block : transactions, hash du block précédent, nonce (proof of work).
* *algorithme de hashage* : formule mathématique qui est appliquée à un nombre variable de données (”input”) en vue de les transformer en un nombre fixe de données correspondant à l’empreinte digitale des données (“output”). Dans le cas de l’algorithme SHA256, la taille du code est toujours 256 bits. Le système du hachage est utilisé dans beaucoup d’autres domaines que le bitcoin pour vérifier facilement que les données intiales (“input”) n’ont pas été changées. En effet deux input différents ne peuvent pas donner un output identique. Par ailleurs, l’algorithme de hachage ne fonctionne que dans un sens et il est impossible de retrouver l’input à partir de l’output.
* *nonce* (bitcoin): nombre arbitraire à trouver, qui ajouté au block permet d’obtenir, après hashage, le résultat demandé par la proof of work. Ce *nonce* est fondamentalement le proof of work. Il est difficile à calculer et facile à vérifier par le reste du réseau
* *règle de consensus*: règle pour valider/sélectionner le bloc qui sera intégré dans la blockchain => proof of work, proof of stake, proof of burn
* *proof of work* : solution à un problème mathématique complexe. Le premier à résoudre le problème n'a plus qu'à en diffuser la preuve => proof of work. Dans le cas de bitcoin, correspond au nonce.
* *proof of stake* : les mineurs sont sélectionnés pour participer au processus de hashage sur la base d’un montant (*stake* = mise) qu’il parie sur leur propre capacité à résoudre ce hashage. Plus on parie gros, plus on a de chance d’être sélectionné.
* *proof of burn* : alternative au system POW (très énergivore) et basé sur le principe que les mineurs autorisés à participer au hashag sont ceux qui auront ‘brûlé’ (= expédié à une adresse blockchain inconnue) le plus grand nombre de tokens.

Il semble que POS et POB sont deux méthodes non pas différentes de POW, mais qui la complètent, dans la mesure où elles limitent le nombre de mineurs autorisés à participer au hashage, sur la base de l’argument de la grande quantité d’énergie ‘inutile’ dépensée par tous les mineurs qui n’ont pas gagné la course POW, mais ont quand même utilisé un grande énergie pour tente d’y parvenir.

1. Différences principales bitcoin – ethereum(blockchain)/ether($crypto)

* objectif = distributed applications/smart contracts (codes IT, langage de programmation interne), pas uniquement $crypto (ether)
* ether = monnaie de règlement sur ethereum (pas de plafond)
* émission ether : initialement via ICO, minage (5 ethers par block)
* gestion/développement par Ethereum Foundation (mais public et décentralisé)
* 1 bloc miné toutes les 12s
* consensus basés sur proof of work mais algo = Ethash et pas SHA256 (sinon fonctionnement similaire)
* fork envisagé avec passage à proof of stake
* récompenses mineurs : ….
* frais de transaction (gaz) : montants définis par type d’opération demandé par l’exécution du code (smart contract)

1. Extensions
2. SmartContracts

….

Smart contracts can:

Function as ‘multi-signature’ accounts, so that funds are spent only when a required percentage of people agree

Manage agreements between users, say, if one buys insurance from the other

Provide utility to other contracts (similar to how a software library works)

Store information about an application, such as domain registration information or membership records.

1. ICO

...

1. Liens divers :

* Transactions crypto

<https://www.blockchains-expert.com/portefeuille-bitcoin-comment-conserver-vos-cryptomonnaies/>

<https://www.blockchains-expert.com/4-etapes-pour-comprendre-une-transaction-bitcoin/>

https://en.bitcoin.it/wiki/Main\_Page

[https://bitcoin.org/en/developer-guide#transactions](https://bitcoin.org/en/developer-guide" \l "transactions)

<https://pouruneautreeconomie.fr/comment-fonctionne-transaction-bitcoin/>

<https://www.les-crises.fr/approfondir-le-bitcoin-deroulement-dune-transaction-23/>

<https://bitcoin.fr/public/divers/Etapes-transaction-bitcoin.jpg>

* Clé privée / publique / transactions

https://medium.com/@JB\_Pleynet/le-chiffrage-%C3%A0-cl%C3%A9s-publiques-priv%C3%A9s-expliqu%C3%A9-aux-non-initi%C3%A9s-1a0eed15934f

<https://en.bitcoinwiki.org/wiki/Pay-to-Pubkey_Hash>

<https://learnmeabitcoin.com/guide/p2pkh>

* Licences bancaires

<https://www.bilan.ch/finance/la-finma-elargit-la-voie-a-la-crypto-avec-deux-licences-bancaires>

[https://www.allnews.ch/content/r%C3%A9glementation/fintech-le-conseil-f%C3%A9d%C3%A9ral-concr%C3%A9tise-la-licence-bancaire-%C2%ABlight%C2%BB](https://www.allnews.ch/content/réglementation/fintech-le-conseil-fédéral-concrétise-la-licence-bancaire-)

* Bitcoin vs autres

https://www.investopedia.com/articles/investing/031416/bitcoin-vs-ethereum-driven-different-purposes.asp

<https://www.crypto-france.com/ethereum-vs-bitcoin/>

<https://www.coindesk.com/learn/ethereum-101/how-ethereum-works>

* Participants

<https://openclassrooms.com/fr/courses/3925766-comprendre-le-bitcoin-et-la-blockchain/4160996-le-reseau-de-mineurs>

* Smart contracts, ICO

1. Différence fondamentale avec no de compte : une nouvelle adresse peut être créée pour chaque nouvelle transactions (conseillé !) [↑](#footnote-ref-2)
2. NB : wallet cryptée ! accès via mot de passe [↑](#footnote-ref-3)
3. Une exception : les blocs orphelins. Lorsque deux mineurs valident un bloc et trouvent le nonce en même temps [ça arrive], il y a deux blocs valides qui sont communiqués au système. On parle alors de ‘fork’ dans la chaine. Le premier des deux blocs à partit duquel est produit un nouveau bloc supplémentaire est accepté, et l’autre bloc est réputé ‘orphelin’ [↑](#footnote-ref-4)