Pas de fausse *blockchain* !

Pour rendre une *blockchain* robuste et impossible à simuler, on utilise une fonction de hachage *h* (publique) qui à tout fichier F associe un code *h*(*F*), une courte suite de caractères dénommée empreinte.

Quand on change un seul caractère de F, le code *h*(F) est totalement changé et ce de façon imprévisible. Il est impossible en pratique de modifier F en F' de manière que *h*(F) = *h*(F') *(voir la rubrique Logique & calcul d'avril 2014).*

On utilise aussi une preuve de travail. C'est une fonction *p* qui, à toute valeur *V* donnée (en général, *V* est une suite de caractères qui détermine un problème), associe une solution *p*(*V*) qui ne s'obtient qu'en menant un long calcul, par exemple d'une durée d'une heure avec une machine de bureau ordinaire. En revanche, vérifier que *p*(*V*) est la bonne valeur associée à *V* est rapide. Calculer le résultat *R* de *p*(*V*) est difficile, mais vérifier que *p*(*V*) = *R* est facile. Un exemple est la décomposition d'un nombre en facteurs premiers, qui est difficile, alors que la vérification que le produit de ces facteurs donne le bon résultat est facile.

Construisons une *blockchain* en concaténant des *blocks* contenant des suites de caractères représentant des messages, ou des transactions : *blockchain* = *block*(1) - *block*(2) - ... - *block*(*n*)

Le *block*(*i*) commence par deux informations particulières *h*(*i*) et *p*(*i*) ajoutées aux messages qui en constituent le contenu. Ces informations *h*(*i*) et *p*(*i*) assurent qu'on ne pourra ni modifier la *blockchain*, ni fabriquer après coup de fausses *blockchains*.

h(*i*) sera l'empreinte de ce qui précède *block*(*i*). Autrement dit, *h*(*i*) = *h*[*block*(1) - ··· - *block*(*i* – 1)].

p(*i*) sera une preuve de travail associée à *h*(*i*) : *p*(*i*) = *p*[*h*(*i*)].

Le calcul des *h*(*i*) et *p*(*i*) se fera progressivement avec l'ajout des *blocks*. Chaque *p*(*i*) exige peut-être une heure de calcul en moyenne, mais résulte d'un calcul collectif qui a pris beaucoup moins qu'une heure par participant. Le calcul de tous les *p*(*i*) se fait progressivement : pour le bitcoin, on a ajouté un *block* toutes les dix minutes depuis janvier 2009 et l'on rétribue ceux qui mènent les calculs.

Quand la *blockchain* comporte de nombreuses pages (la *blockchain* du bitcoin en a environ 300 000), faire une fausse *blockchain* est impossible, car cela demande trop de calculs. Pour 300 000 pages « coûtant » chacune une heure de calcul, il faudrait faire un calcul de 300 000 heures, soit 34 années (et pour le bitcoin, il existe d'autres garde-fous qui font que le calcul est bien plus long). Cependant, vérifier que la *blockchain* est cohérente est facile :

1) On vérifie que chaque *h*(*i*) est convenable en recalculant sa valeur, ce qui est possible car tout est public. C'est rapide et cela assure que rien n'a été modifié dans les *blocks*. 2) On vérifie les *p*(*i*), ce qui est rapide aussi.

Une telle *blockchain* est un objet numérique que chacun peut contrôler rapidement, mais que personne ne peut modifier sans se faire repérer et que personne ne peut imiter, car pour cela il faudrait mener un énorme calcul.

© YinYinPhoto/shutterstock.com

de trop gros volumes de données), mais seulement les informations d'enregistrement et d'authentification. Les messages sont gérés par un système de dépôt distribué (c'est-à-dire répartis entre les utilisateurs) avec une table permettant de savoir où se trouvent les informations, comme cela se pratique dans les réseaux pair à pair de partage de musique.

Un projet plus ambitieux, car se voulant le support possible d'applications complexes fondé sur une notion de contrat *(smart-contract)*, est en cours de développement : il se nomme *Ethereum*. La *blockchain* associée émettra une monnaie (des *éthers*) sur le modèle du bitcoin, mais ce ne sera qu'une de ses fonctions.

Relier les chaînes

Une autre avancée toute récente dans ce domaine a été proposée par un groupe de chercheurs associé à Adam Back, inventeur britannique d'un système de preuve de travail *(voir cette rubrique dans Pour la Science d'avril 2014)* utilisé comme élément du protocole du bitcoin. Adam Back et ses collègues ont constaté que le bitcoin évolue très lentement, les décisions pour tout changement se faisant selon un processus où il faut un accord difficile à obtenir de la part de ceux qui travaillent à le surveiller et qui ne sont pas organisés en structure hiérarchique : c'est un problème avec les applications totalement décentralisées dont le contrôle n'est aux mains de personne.

Adam Back a aussi noté que beaucoup d'idées innovantes proposées par des *blockchains* nouvelles n'ont qu'un succès limité ; en valeur, le bitcoin reste très dominant parmi les monnaies cryptographiques. Lui et ses collègues ont donc mis au point une méthode liant les *blockchains*. Ce système de *sidechain* permettra de faire passer des unités monétaires d'une chaîne A vers une autre, B. Elles disparaîtront de la chaîne A pour réapparaître sur B et pourront éventuellement revenir dans A.

Chaque *blockchain* est un petit univers où il est utile de disposer d'une monnaie (comme sur *Namecoin*) ; cependant, faire accepter une nouvelle monnaie et stabiliser

Mieux que *Leboncoin*, *eBay*, *Priceminister*, etc.

Le commerce entre particuliers sur Internet est freiné par un problème de confiance. Vous mettez en vente un magnifique vase, vous trouvez un acheteur, vous lui envoyez l'objet... et il ne vous paye jamais – ou, à l'inverse, il vous envoie l'argent et vous gardez le vase...

Même si c'est malhonnête, il est économiquement rationnel pour celui qui reçoit l'envoi de son correspondant de ne pas envoyer ce qu'il a promis en échange. *eBay* précise d'ailleurs : « Nous ne pouvons pas obliger le vendeur à remplir ses obligations. »

Leboncoin recommande l'échange en direct avec rencontre. L'envoi contre remboursement est une solution et certains sites (parfois associés à *Paypal*) proposent leurs services pour limiter les risques. Soit ils bloquent l'argent envoyé avant que le produit ne soit reçu ; soit ils proposent un système de notation qui indique si les personnes avec qui on fait affaire ont été correctes dans leurs précédentes transactions ; soit ils proposent une assurance.

Il existe pourtant une meilleure solution grâce aux *blockchains*. Oleg Andreev a proposé un système inspiré par la *blockchain* du bitcoin, qui résout le problème sans frais. L'idée s'applique à toute *blockchain* disposant d'un système d'unités monétaires permettant les transactions à plusieurs entrées et plusieurs sorties (c'est le cas de la *blockchain* du bitcoin).

Supposons qu'Alain veuille vendre un vase à 100 euros à Béatrice qui habite loin. Ils sont d'accord sur le prix, mais doivent faire l'échange à distance. Alain et Béatrice, indépendamment des 100 euros convenus, déposent chacun 200 euros sur un compte particulier. Lorsque l'échange sera terminé (envoi du vase et envoi des 100 euros pour le payer), Alain et Béatrice signeront la transaction qui débloquera les 400 euros, lesquels seront alors restitués : 200 pour Alain, 200 pour Béatrice.

La procédure de mise sous séquestre sur la *blockchain* est telle que personne ne peut s'emparer de cet argent, sauf Alain et Béatrice s'ils donnent tous deux leur accord. Seul, aucun d'eux ne peut rien faire. En conséquence, à moins d'être prêt à perdre 200 euros pour un objet qui en vaut 100, Béatrice aura intérêt à payer le vase. De même, si Béatrice paye avant de recevoir le vase, Alain aura intérêt à envoyer le vase (qui vaut 100 euros) pour récupérer ses 200 euros bloqués. Il est ainsi économiquement rationnel de se comporter honnêtement !

La somme de 200 euros pour une transaction concernant un objet qui en vaut 100 est modifiable, mais il faut que la somme mise sous séquestre par les deux acteurs soit supérieure à la valeur de l'objet échangé. Sinon, celui qui reçoit l'envoi de l'autre en premier aura intérêt à ne pas envoyer ce qu'il doit, quitte à perdre la somme séquestrée. Ce système à base de *blockchain* permet, sans l'action d'aucune autorité centrale, de mener une opération d'échange avec un risque très réduit de se faire escroquer (pour des précisions sur la mise en œuvre technique, voir http://bit.ly/1BuoupD ou http://voluntary.net/bitmarkets/).

son cours est difficile et incertain. De plus, chaque *blockchain* est une expérience aux risques d'autant plus grands qu'elle est récente et innovante. Une fois mis en place, le système des *sidechains* (ce n'est pas si simple et, aujourd'hui, aucune *sidechain* ne fonctionne) permettra de tester rapidement de nouvelles idées. Chacune pourra « importer » la monnaie d'une autre *blockchain*, sans doute le bitcoin, la monnaie la mieux installée et celle pour laquelle la confiance est la plus forte. Le système est conçu pour que la chaîne qui « prête » de l'argent à une autre n'engage pas plus que ce qu'elle prête ; elle ne prend donc pas de risque.

Avec un tel système, non seulement les diverses *blockchains* ne se concurrenceront plus nécessairement, toutes pouvant s'appuyer sur une seule ou un petit nombre d'entre elles, mais de plus toutes les expérimentations pourront être envisagées sans crainte, créant une dynamique propice aux innovations.

Plutôt que d'expliquer les architectures complexes et spécifiques de *Namecoin, Twistter* ou *Ethereum*, terminons en présentant les applications générales les plus simples d'une *blockchain*.

Des outils cryptographiques

Une *blockchain* est un fichier numérique accessible à tous en lecture et en écriture. Cela se fait par l'utilisation de logiciels parfois nommés *servents,* terme qui associe les deux mots *client* et *serveur* utilisés dans les réseaux centralisés. Ce mot marque que, dans un réseau pair à pair, chaque nœud est à la fois nœud central (serveur) et utilisateur (client).

Le fichier *blockchain* est rendu infalsifiable et indestructible par l'utilisation d'outils cryptographiques. Quand on ajoute quelque chose au fichier (on parle de *page* ou de *block*), on le fait en commençant l'ajout par un code calculé à partir de la version avant ajout (pour calculer ces codes, on utilise des fonctions de hachage). On ne pourra donc pas modifier la partie ancienne de la *blockchain* sans que cela se voie, ou alors il faudrait modifier chacun des codes commençant les pages : les codes rigidifient le fichier.

De plus, pour qu'il soit difficile de fabriquer une *blockchain* entièrement artificielle avec de bons codes en tête de chaque page ajoutée et qui prétendrait se substituer à la vraie *blockchain*, le protocole de gestion exige qu'on fasse un certain travail de calcul pour ajouter une page ; c'est la notion de *preuve de travail (voir l'encadré page 83)*. Par conséquent, fabriquer une fausse *blockchain*, ou même seulement une fausse partie finale de *blockchain*, est excessivement coûteux, voire impossible, en pratique. Au total, le fichier *blockchain* ne peut être ni modifié ni simulé. Comme la *blockchain* existe en une multitude de