L'empire romain ne doit pas être géré comme une petite île grecque

Emmanuelle Anceaume, Romaric Ludinard, Bruno Sericola

L'empire romain ne doit pas être géré comme une petite île grecque

Emmanuelle Anceaume[1], Romaric Ludinard[2], Bruno Sericola[3]

[1]*CNRS, UMR 6074 - IRISA, emmanuelle.anceaume@irisa.fr*
[2]*ENSAI, UMR 9194 - CREST, romaric.ludinard@ensai.fr*
[3]*INRIA Rennes - Bretagne Atlantique, bruno.sericola@inria.fr*

Depuis quelques années, des systèmes de crypto-devises ont fait leur apparition, permettant d'acheter des biens ou services de manière décentralisée en s'affranchissant de tout système bancaire. Bitcoin est un système pair-à-pair de crypto-devises considéré comme l'un des pionniers de ce type de système. Un adversaire peut chercher à mettre en défaut le système en insérant des données incohérentes dans l'historique des transactions, appelé blockchain. En exploitant une situation de compétition, il peut alors acquérir des biens ou services gratuitement. Afin de bloquer ces comportements, PeerCensus propose d'utiliser des techniques d'accord basées sur les identités présentes dans la blockchain. Dans cet article, nous proposons une évaluation probabiliste de la fiabilité de l'architecture PeerCensus.

Mots-clefs : Bitcoin, Système large échelle, Tolérance aux fautes, Analyse probabiliste

1 Introduction

Assez naturellement, le problème de la construction d'un système de paiement entre utilisateurs ne s'appuyant pas sur une banque s'est posé. Un tel système doit fournir deux fonctionnalités de base : authentification et non-répudiation. L'authentification permet de s'assurer de la légitimité d'un utilisateur à effectuer un paiement tandis que la non-répudiation permet d'empêcher toute remise en cause d'un paiement effectué. L'utilisation de protocoles cryptographiques permet d'assurer ces propriétés.

2 Le système de crypto-devises Bitcoin

Bitcoin [Nak08] est un système de crypto-devises introduit en 2008 par Nakamoto mettant en œuvre ces fonctionnalités. Le système Bitcoin repose sur deux opérations : la création de devises bitcoin et l'utilisation de ces devises. L'utilisation des bitcoins est faite au travers de *transactions*, $T_t, \ldots, T_{t'}$, chacune de ces transactions identifiant l'acheteur, le vendeur, la valeur du bien ou du service acheté et une preuve de solvabilité de l'acheteur. Plus précisément, un client c souhaitant acheter un bien d'une valeur m auprès d'un vendeur v, crée une transaction $T_{t'} = (\{T_{t_1}, \ldots, T_{t_\ell}\}, c, v, m)$ où $\{T_{t_1}, \ldots, T_{t_\ell}\}$ représentent la preuve de solvabilité de $T_{t'}$ (à savoir $\{T_{t_1}, \ldots, T_{t_\ell}\}$ sont des transactions passées au cours desquelles c était vendeur (i.e., $T_{t_1}.v = \ldots = T_{t_\ell}.v = T_{t'}.c$) et la quantité de bitcoins reçu par c est suffisante pour l'achat courant (i.e., $T_{t_1}.m + \ldots + T_{t_\ell}.m \geq T_{t'}.m$). Le client c signe cryptographiquement cette transaction avant de la transmettre au vendeur v. L'utilisation de la signature cryptographique permet d'assurer que la transaction provient bien du client. Concernant la solvabilité du client, le vendeur interroge le système Bitcoin afin de vérifier que le client n'ait pas créé des transactions dont le montant invaliderait la transaction $T_{t'}$. Cette vérification est faite grâce à la *blockchain*. La blockchain correspond au grand livre des comptes du système Bitcoin. Elle contient toutes les transactions effectuées depuis l'origine du système. Chaque pair participant au système Bitcoin maintient une copie locale de la *blockchain*, et donc peut immédiatement vérifier la solvabilité d'un client en agrégeant les montant perçus et dépensés par ce client depuis l'origine du système. Si la vérification est positive (on parle alors de transaction *valide*), le vendeur vend son produit au client et attend l'enregistrement de la transaction dans la blockchain pour l'encaissement de sa vente. Notons que les pairs du système Bitcoin sont organisés suivant un graphe aléatoire. Lorsqu'un pair valide une transaction, il la

diffuse à ses voisins. La transaction est ensuite propagée de proche en proche jusqu'à ce que tous les pairs du système aient connaissance de cette transaction. Les transactions ne sont pas immédiatement enregistrées dans la blockchain, mais sont temporairement mises en attente localement jusqu'à ce que le pair crée un bloc. La création d'un bloc repose sur la création d'une preuve de travail qu'il est difficile d'obtenir d'un point de vue calculatoire mais dont la vérification est facile. Cette preuve de travail dépend de l'état courant de la blockchain, et aujourd'hui, une preuve de travail est trouvée toutes les 10 minutes en moyenne. Dès qu'un pair a réussi à générer une preuve de travail, il construit un bloc en y insérant toutes les transactions en attente localement depuis le dernier bloc de la blockchain, et diffuse ce bloc dans le système Bitcoin. La blockchain revient donc à une séquence ordonnée temporellement de blocs. Les transactions qui étaient en attente sont donc enregistrées de manière irrémédiable dans la blockchain. La création d'un bloc est rémunérée par le système, créant ainsi une incitation pour les pairs du système à créer des blocs.

3 L'attaque par "double spending"

Le système Bitcoin est vulnérable aux attaques de type double spending. Ces attaques consistent à générer au moins deux transactions faisant référence à une même transaction antérieure et telle que la somme de leur montant excède celui de la transaction antérieure. La première de ces transactions, à destination d'un vendeur légitime, est soumise par ce dernier dans le système Bitcoin pour vérifier la solvabilité du client, et la seconde, à destination du client lui-même, est envoyé par ce dernier à un autre endroit du système (rappelez vous que tous les pairs du système Bitcoin peuvent vérifier la solvabilité des transactions en s'appuyant sur leur copie locale de la blockchain). Si un bloc est généré par un pair ayant uniquement connaissance de la seconde transaction, la première n'ayant pas eu le temps de traverser le système, alors seule la seconde transaction sera enregistrée dans la blockchain. La première, incompatible avec la seconde (l'antériorité n'étant plus valide), ne sera jamais insérée dans la blockchain et donc le client aura obtenu son produit sans payer un bitcoin.

Cette attaque a été identifiée très tôt par les concepteurs du système Bitcoin. Certaines contre-mesures existent permettant de diminuer les chances de réussite de cette attaque, mais elle reste néanmoins une faiblesse du système. Cette attaque exploite une situation de compétition dans le protocole Bitcoin : la vérification et l'enregistrement d'une transaction ne constituent pas une séquence atomique d'opérations. De ce fait, le modèle de cohérence garanti par le protocole n'est pas suffisant pour empêcher ce type d'attaque.

4 L'approche PeerCensus

Les auteurs de [DSW16] proposent d'améliorer le système Bitcoin, en particulier sa résistance aux attaques par double spending, en garantissant une cohérence atomique entre différentes opérations du système, à savoir la vérification et l'enregistrement d'une transaction. La cohérence atomique assure que toutes les mises à jours effectuées sur un objet dans le système sont perçues dans le même ordre par toutes les entités du système. PeerCensus repose sur deux concepts fondamentaux : tout d'abord l'indépendance entre les processus de génération de blocs et d'enregistrement des transactions et ensuite l'utilisation d'algorithmes de consensus [Lam98] résilients aux fautes byzantines tels que [KAD+07, CL99]. Dans [DSW16], l'état courant des débits (des clients) et des crédits (des vendeurs) est maintenu grâce à des algorithmes de consensus exécutés par l'ensemble $\mathcal{E}$ des pairs ayant réussi à générer au moins un bloc depuis l'origine du système. Les pairs de $\mathcal{E}$ s'accordent également, au moyen de consensus, sur l'extension de $\mathcal{E}$ aux nouveaux pairs ayant réussi à créer un bloc.

D'un point de vue complexité algorithmique, cette approche est fortement tributaire de la popularité du système Bitcoin. En effet, à chaque création d'un nouveau bloc dans la blockchain, le nombre de participants de $\mathcal{E}$ impliqués dans la prochaine exécution d'un consensus peut augmenter d'une unité. Aujourd'hui, la blockchain comprend environ $k = 395000$ blocks, donc potentiellement 395000 † pairs sont impliqués dans l'exécution de chaque consensus (la complexité en nombre de messages des algorithmes de consensus tolérants les byzantins est typiquement de k^3).

†. Une coincidence étonnante : 395000 correspond au nombre de citoyens romains sous la république en l'an -125

Au-delà de cet aspect, d'un point de vue algorithmique, relier la composition du groupe participant à la k-ème exécution du consensus à la décision prise lors de la $(k-1)$-ème exécution conduit avec forte probabilité à la corruption du groupe $\mathcal{E}$ (i.e., à la présence de plus d'un tiers de pairs byzantins participant à la k-ème execution du consensus) et ce, même si la proportion de byzantins dans le système Bitcoin ne dépasse pas un tiers (la présence d'au plus d'un tiers de participants byzantins est une condition nécessaire pour assurer la sûreté d'un accord [LSP82]). La suite de cet article dérive cette probabilité de corruption.

Parmi tous les pairs cherchant à créer des blocs dans le système, on désigne par $\mu \in]0,1[$ la proportion de pairs byzantins. Dans la suite un pair non byzantin est dit correct. Nous supposerons également que le temps s'écoulant entre la génération de deux blocs consécutifs est constant. Soit $B_k = (h,m)$ l'état de la blockchain à l'instant k, où h désigne le nombre de blocs générés par les pairs corrects et m le nombre de blocs générés par les pairs byzantins. On suppose que le concepteur du système Bitcoin est correct et donc on a $B_0 = (1,0)$.

Le processus $B = \{B_k,\ k \geq 0\}$ avec $B_k \in \mathbb{N}^* \times \mathbb{N}$ représente l'évolution de la blockchain au cours du temps. Lorsque la blockchain est dans l'état (h,m), seules deux transitions sont possibles : soit le bloc généré provient d'un pair correct, auquel cas la blockchain passe dans l'état $(h+1,m)$ avec probabilité $1-\mu$, soit ce bloc est généré par un pair byzantin et la blockchain passe dans l'état $(h,m+1)$ avec probabilité μ. Le processus B est une chaîne de Markov à temps discret sur l'espace d'états $\mathbb{N}^* \times \mathbb{N}$. Les transitions de probabilité non nulles de la matrice de transition P du processus B sont données, pour tout $(h,m) \in \mathbb{N}^* \times \mathbb{N}$, par $P_{(h,m),(h+1,m)} = 1-\mu$ et $P_{(h,m),(h,m+1)} = \mu$.

L'état $B_k = (h,m)$ de la blockchain à l'instant k est dit corrompu si le nombre m de pairs byzantins participants à la k-ème exécution du consensus est supérieur ou égal à $(k-1)/3$. On partitionne l'espace $\mathbb{N}^* \times \mathbb{N}$ en deux sous-ensembles $\mathcal{S}$ et $\mathcal{P}$ correspondant respectivement à l'ensemble des états sains et l'ensemble des états corrompus de la blockchain. On a donc

$$\mathcal{S} = \{(h,m) \in \mathbb{N}^* \times \mathbb{N} \mid h \geq 2m+1\} \text{ et } \mathcal{P} = \{(h,m) \in \mathbb{N}^* \times \mathbb{N} \mid h \leq 2m\}.$$

On obtient facilement, pour tout $h \geq 1$, $m \geq 0$ et $k \geq 0$, $\Pr\{B_k = (h,m)\} = \binom{m+h-1}{h-1}(1-\mu)^{h-1}\mu^m 1_{\{k=m+h-1\}}$, avec $(1,0)$ l'état initial de la chaîne. On obtient alors

$$\Pr\{B_k \in \mathcal{S}\} = \sum_{h=1,\ 3h \geq 2k+3}^{k+1} \binom{k}{h-1}(1-\mu)^{h-1}\mu^{k-h+1} = \sum_{h=\lceil 2k/3 \rceil}^{k} \binom{k}{h}(1-\mu)^h \mu^{k-h}. \tag{1}$$

En utilisant le théorème central limite, on obtient

$$\lim_{k \longrightarrow \infty} \Pr\{B_k \in \mathcal{S}\} = \begin{cases} 0 & \text{si} \quad \mu > 1/3 \\ 1/2 & \text{si} \quad \mu = 1/3 \\ 1 & \text{si} \quad \mu < 1/3. \end{cases} \tag{2}$$

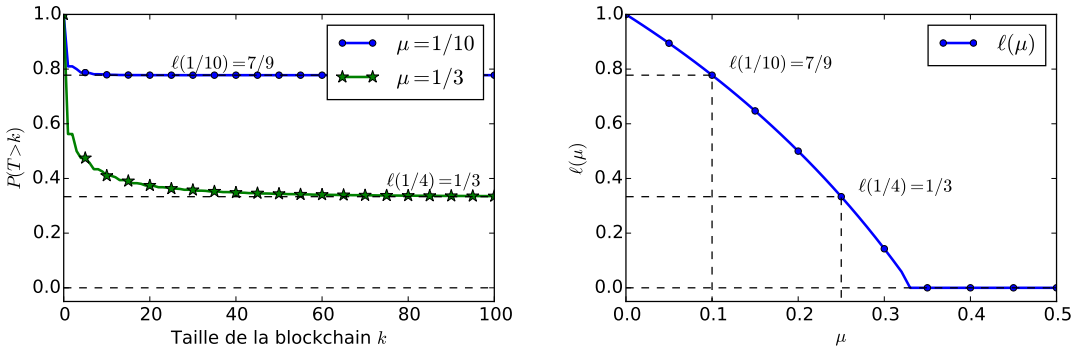
Ce résultat est en accord avec celui présenté dans [DSW16]. Il ne permet pas d'affirmer néanmoins que toutes les exécutions de consensus qui ont mené à l'état B_k sont correctes, *i.e.*, pour tout $k' \leq k$, $B_{k'} \in \mathcal{S}$. Soit T le temps passé dans le sous-ensemble $\mathcal{S}$ avant d'atteindre pour la première fois un état de $\mathcal{P}$, c'est-à-dire la durée pendant laquelle toutes les exécutions du consensus sont correctes. La variable aléatoire T est définie par $T = \min\{k \geq 0 \mid B_k \in \mathcal{P}\}$ et on a $\Pr\{T > k\} = \Pr\{B_0 \in \mathcal{S}, B_1 \in \mathcal{S}, \ldots, B_k \in \mathcal{S}\}$. Le théorème 1 donne la loi du premier instant de corruption du consensus. Nous exhibons également le comportement asymptotique de cette loi.

Théorème 1 *Étant donné $0 < \mu < 1$, pour tout $k \geq 0$, on a*

$$\Pr\{T > k\} = \frac{1}{1-\mu} \sum_{h=\lceil 2k/3 \rceil + 1}^{k+1} \binom{k+1}{h}(1-\mu)^h \mu^{k+1-h} - \frac{3\mu}{1-\mu} \sum_{h=\lceil 2k/3 \rceil + 1}^{k} \binom{k}{h}(1-\mu)^h \mu^{k-h}. \tag{3}$$

De plus, on a :

$$\ell(\mu) = \lim_{k \longrightarrow \infty} \Pr\{T > k\} = \begin{cases} 0 & si \quad \mu > 1/3 \\ 1 - \dfrac{2\mu}{1-\mu} & si \quad \mu \leq 1/3. \end{cases} \tag{4}$$



(a) Évaluation de $\Pr(T > k)$ pour différentes valeurs de μ en fonction de la taille de la blockchain.

(b) Comportement asymptotique de $\Pr(T > k)$ en fonction de la proportion μ de byzantins.

FIGURE 1: Loi du temps de séjour dans les états sains.

La figure 1(a) illustre la loi de T pour deux valeurs de μ, $\mu = 1/10$ et $\mu = 1/3$. On constate que la loi converge très rapidement vers la limite $\ell(\mu)$. La figure 1(b) illustre quant à elle le comportement asymptotique de T pour différentes proportions de byzantins ($0 < \mu < 1$). Pour une proportion $\mu > 1/3$ de byzantins, on retrouve bien le résultat classique de [LSP82]. En revanche, la probabilité d'avoir une suite de consensus corrects en présence de $\mu < 1/3$ byzantins est strictement inférieure à 1. Cela peut sembler surprenant et en contradiction avec la relation 2. Néanmoins il n'en n'est rien car la relation 2 n'exprime que la probabilité d'être dans un état sain à l'instant k tandis que le Théorème 1 donne la probabilité de n'être passé que dans des états sains jusqu'à l'instant k. Par exemple pour $\mu = 1/4$, cette probabilité tend vers $\ell(1/4) = 1/3$.

5 Pour conclure

Cette étude fournit une analyse fine des garanties fournies par l'exécution d'une séquence de consensus en présence d'entités byzantines. Si la valeur décidée par un ensemble $\mathcal{E}_k$ lors de la k-ème exécution n'influence pas la composition de $\mathcal{E}_{k+1}$, alors les bornes de sûreté de [LSP82] sont vraies pour tout k. Dans le cas contraire même la présence de moins d'un tiers de byzantins ne permet pas de garantir un succession d'exécutions correctes, remettant ainsi en cause la correction de l'approche présentée dans [DSW16]. Moralité, cette étude montre que même si le parlement est le "destin de la démocratie" (théorie prônée par Hans Kelsen), il ne doit en aucun cas décider de sa propre constitution.

Références

[CL99] M. Castro and B. Liskov. Practical Byzantine Fault Tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI)*, 1999.

[DSW16] C. Decker, J. Seidel, and R. Wattenhofer. Bitcoin Meets Strong Consistency. In *17th International Conference on Distributed Computing and Networking (ICDCN), Singapore*, 2016.

[KAD+07] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong. Zyzzyva : Speculative byzantine fault tolerance. In *In Symposium on Operating Systems Principles (SOSP)*, 2007.

[Lam98] L. Lamport. The Part-time Parliament. *ACM Trans. on Computer Systems*, 16(2), 1998.

[LSP82] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3), 1982.

[Nak08] S. Nakamoto. Bitcoin : A Peer-to-Peer Electronic Cash System. `https://bitcoin.org/bitcoin.pdf`, 2008.