**Systèmes d’information avancés. Projet TP.**

**Sergey Kirgizov**

**ESIREM**

**Tchaî — Chaîne des transactions**

**Objectif : concevoir un système de transactions électroniques avec une intégrité garantie, accessible**

**par le protocole HTTP.**

**Mode du travail préférée : binômes.**

**À la ﬁn de ce document, vous trouverez des liens vers la documentation. Cela vous aidera à eﬀectuer les**

**exercices suivants. Après avoir terminé tous les exercices, vous obtiendrez un système de gestion des**

**transactions ressemblant à celui de la blockchain et git.**

****

****

**EXERCICE 1. Premiers pas de git.**

**1. Créer un** [**dépôt**](#br3) **git dans un répertoire existant local (git init).**

**2. Rédiger un** [**ﬁchier**](#br3) **README.md contenant la description du projet en utilisant un langage de balisage**

**Markdown** [**[2].**](#br3) **Ajouter ce ﬁchier dans le dépôt (git add).**

**3. Faire le premier git commit.**

**4. Consulter le statut du dépôt (git status) et la liste des commits (git log).**

**EXERCICE 2. Git, un outil de** [**travail collaboratif.**](https://bitbucket.org)

**1. Créer un dépôt privé sur** [**https://bitbucket.org**](https://bitbucket.org)**.**

**Nommer ce dépôt comme TCHAI-PRENOM -NOM -PRENOM -NOM en utilisant vos noms et prénoms.**

**1**

**1**

**2**

**2**

**2. Synchroniser (git push) votre dépôt local avec le dépôt que vous venez de créer sur bitbucket.**

**3. Ajouter votre binôme et votre enseignant en tant que collaborateurs dans le projet sur bitbucket.**

**4. Sur l’ordinateur du binôme :**

**(a) Cloner le dépôt de bitbucket sur l’ordinateur du binôme.**

**(b) Modiﬁer le ﬁchier README.md en ajoutant une nouvelle section “Auteurs” contenant vos noms**

**et les adresses mails.**

**(c) Enregistrer des modiﬁcations dans le dépôt local (git add, puis git commit)**

**(d) Pousser son travail sur le dépôt distant de bitbucket (git push)**

**5. Sur le premier ordinateur :**

**(a) Récupérer les modiﬁcations de votre binôme, en tirant depuis le dépôt distant (git pull)**

**(b) Vériﬁer que tout va bien en consultant le statut du dépôt (git status) et la liste des commits**

**(git log).**

**1**



**Par défaut, Git vous permet de partager un dépôt privé avec au plus trois personnes, donc le travaille**

**en trinôme peut être problématique. Au cours du projet, vous devez synchroniser régulièrement avec**

**bitbucket, sans oublier de mettre à jour la description du projet et la documentation dans le ﬁchier**

**README.md. Chaque version signiﬁcative de votre code doit être étiquetée (git tag). Votre enseignant va**

**regarder votre code de temps en temps aﬁn de vous donner les commentaires constructifs.**

****

**INFO : Pour chaque exercice terminé vous devez avoir au moins un commit correspondant dans le**

**dépôt git. Le nombre de commits ne doit pas être inférieur au nombre d’exercices résolus.**

**Tchaî v1**

**Nous déﬁnissons une transaction** [**comme**](#br3) **étant un tuplet (P , P , t, a), où a est égal à la somme d’argent**

**1**

**2**

**transférée de la personne P à** [**la personne**](#br3) **P au moment t.**

**1**

**2**

****

****

**EXERCICE 3. En utilisant Flask** [**[3],**](#br3) **réaliser une première version du système “Tchaî”. Voici une liste des**

**actions qui doivent être mises à la disposition via un API HTTP (voir TD-1) par votre système “Tchaî” :**

**(A1) Enregistrer une transaction.**

**(A2) Aﬃcher une liste de toutes les transactions dans l’ordre chronologique.**

**(A3) Aﬃcher une liste des transactions dans l’ordre chronologique liées à une personne donnée.**

**(A4) Aﬃcher le solde du compte de la personne donnée.**

**INFO : Vous êtes libre de choisir un autre langage de programmation au lieu du** [**Python,**](#br3) **un autre**

**framework web au lieu de Flask, un système préféré de stockage de données (SQLite** [**[8],**](#br3) **un ﬁchier**

**texte, ...), etc. Vos choix doivent être documentés et justiﬁé dans le ﬁchier README.md.**

****

****

**EXERCICE 4. Attaquer le système en modiﬁant directement le ﬁchier de données, en changeant le**

**montant d’une transaction.**

**INFO : Toutes les attaques et tous les tests doivent être décrits en détail dans une section adéquate**

**du ﬁchier README.md. Les scripts permettant leur exécution doivent être disponibles dans le répertoire**

**tests .**

**Tchaî v2**

**Nous ajoutons maintenant le hash d’une transaction dans son tuplet : (P , P , t, a, h), où a est égal à la**

**1**

**2**

**somme d’argent transférée de la personne P à la personne P au moment t et h correspond au hash**

**1**

**2**

**du tuple (P , P , t, a).**

**1**

**2**

****

**INFO : Votre choix de la fonction de hachage doit également être documenté dans le ﬁchier README.md.**

****

****

**EXERCICE 5. Modiﬁer votre programme aﬁn d’intégrer la nouvelle structure des transactions**

**EXERCICE 6. Ajouter l’action suivante disponible en API HTTP :**

**(A5) Vériﬁer l’intégrité des données en recalculant les hashs à partir des données et en les comparant**

**avec les hashs stockés précédemment.**

****

****

**EXERCICE 7. Vériﬁez que l’attaque précédente ne fonctionne plus.**

**EXERCICE 8. Attaquer le [système](#br3) en modiﬁant directement le ﬁchier de données, en supprimant une**

**transaction. La possibilité de [supprimer](#br3) une transaction peut être très dangereuse, la suppression peut**

**entraîner la double dépense [[9]](#br3)**

**2**

**Tchaî v3**

****

**EXERCICE 9. Modiﬁer la méthode de calcul de hash. Maintenant la valeur du hash h**

**va dépendre**

**i+1**

**non seulement de la transaction en cours, mais également de la valeur du hash h de la transaction**

**i**

**précédente.**

****

****

**EXERCICE 10. Vériﬁez que les attaques précédentes ne fonctionnent plus.**

**EXERCICE 11. Attaquer le système en modiﬁant directement le ﬁchier de données, en ajoutant, par**

**exemple, une transaction provenant d’une autre personne vers le compte de l’attaquant.**

**Tchaî cryptogr**[**aphique**](#br3) **(v4)**

****

****

**EXERCICE 12. Lire le message** [**[4],**](#br3) **le papier original de Satoshi Nakamoto** [**[5]**](#br3) **et la discussion ultérieure**

**sur la liste de diﬀusion ‘The Cryptography and Cryptography Policy Mailing List”.**

**EXERCICE 13. Utiliser la cryptographie asymétrique aﬁn d’assurer l’authenticité de l’expéditeur.**

[**Références**](https://git-scm.com/book/fr/v2)

**[1]** [**Pro Git book, Scott Chacon et Ben**](https://git-scm.com/book/fr/v2) **Straub, 2014.**

[**https://git-scm.com/book/fr/v2**](https://git-scm.com/book/fr/v2)

**[2]** [**Markdown, un langage de balisage léger créé par**](https://daringfireball.net/projects/markdown/) **John Gruber et Aaron Swartz en 2004.**

[**https://daringfireball.net/projects/markdown/**](https://daringfireball.net/projects/markdown/)

**[3]** [**Flask, un framework léger de développement**](https://palletsprojects.com/p/flask/) **web en Python, créé initialement par Armin Ronacher**

[**comme étant un poisson d’avril, le 1er avril**](https://palletsprojects.com/p/flask/) **2010.**

[**https://palletsprojects.com/p/flask/**](https://palletsprojects.com/p/flask/)

**[4] Bitcoin** [**P2P e-cash paper, un message de Satoshi Nakamoto posté sur “The Cryptography**](https://www.metzdowd.com/pipermail/cryptography/2008-October/014810.html) **and**

**Cryptogr**[**aphy Policy Mailing List”, 2008.**](https://www.metzdowd.com/pipermail/cryptography/2008-October/014810.html)

**Archive 1 :** [**https://www.metzdowd.com/pipermail/cryptography/2008-October/014810.html**](https://www.metzdowd.com/pipermail/cryptography/2008-October/014810.html)

[**Archive 2 :**](http://www.bitcoin.org/bitcoin.pdf) [**https://www.mail-archive.com/cryptography@metzdowd.com/msg09959.html**](https://www.mail-archive.com/cryptography@metzdowd.com/msg09959.html)

**[5]** [**Bitcoin : A Peer-to-Peer Electronic Cash**](http://www.bitcoin.org/bitcoin.pdf) **System, Satoshi Nakamoto, 2008**

[**http://www.bitcoin.org/bitcoin.pdf**](http://www.bitcoin.org/bitcoin.pdf)

**[6]** [**Python IDE “Thonny”**](https://thonny.org)**, créé par Aivar Annamaa**

[**https://thonny.org**](https://thonny.org)

**[7]** [**Installation de Git**](https://git-scm.com/download)

[**https://git-scm.com/download**](https://git-scm.com/download)

**[8]** [**SQLite, une bibliothèque**](https://sqlite.org) **écrite en C, proposant un moteur de base de données relationnelle**

[**accessible par le langage**](https://sqlite.org) **SQL. créée par D. Richard Hipp en 2010**

[**https://sqlite.org**](https://sqlite.org)

**[9]** [**Double dépense, (en anglais, double-spending), un acte**](https://fr.wikipedia.org/wiki/Double_d%C3%A9pense) **frauduleux dans lequel le même jeton**

[**numérique est dépensé plus d’une fois**](https://fr.wikipedia.org/wiki/Double_d%C3%A9pense)

[**https://fr.wikipedia.org/wiki/Double\_d%C3%A9pense**](https://fr.wikipedia.org/wiki/Double_d%C3%A9pense)

**3**