Regular Paper

Communications of the CCISA
Vol. 25 No. 4 Nov. 2019

DBL—可否認的區塊鏈帳本

陳炫豪[1*]、紀博文[2]
國立臺灣師範大學資訊工程學系
{60647079s、neokent}@gapps.ntnu.edu.tw

摘要

本文建構了一個可否認的區塊鏈帳本，賦予原本具有不可修改性的區塊鏈可否認的性質，讓此區塊鏈帳本的擁有者能夠藉由產生合法且無法被識破的假交易區塊複本來否認原始的區塊鏈帳本，以防惡意的使用者對帳本的擁有者進行暴力脅迫，也讓區塊鏈上的內容能夠被修改及遺忘。可否認的區塊鏈帳本架構利用了區塊鏈的型式以及 Chameleon Hash 的雜湊碰撞來建構成一個可以修改交易紀錄但仍舊保有相同的雜湊函數值的區塊鏈，透過重複使用雜湊碰撞並產生新區塊的演算法來提供其他使用者假的交易紀錄複本，雖然其他使用者可以發現每份複本皆不相同，但由於不同的區塊前後的雜湊函數值皆合法，所以他們無法證明哪一份複本為真、哪一份複本為假。我們也介紹了兩種使用可否認的區塊鏈帳本的情況，分別是交易紀錄相關的使用者皆為一般使用者的情況以及交易紀錄相關的使用者為惡意使用者之同夥的情況，並提供兩種演算法來應用相對應的狀況。最後我們目前也在進行此架構的實作及測試，期望未來這個可否認的區塊鏈帳本能夠應用到現實社會。

關鍵詞：區塊鏈、可否認性、Chameleon Hash

* 通訊作者 (Corresponding author.)

Communications of the CCISA
Vol. 25 No. 4 Nov. 2019

DBL: Deniable Blockchain Ledger

Hsuan-Hao Chen[1†]、Po-Wen Chi[2]
Department of Computer Science and Information Engineering, National Taiwan Normal University
{ 60647079s、neokent } @gapps.ntnu.edu.tw

Abstract

In this paper, we construct a deniable blockchain ledger (DBL), which enhances blockchain with a new feature, deniability. The owner of the deniable blockchain ledger can produce a new blockchain copied from the original blockchain and replace some blocks with fake blocks. So, the original transaction data are hidden and our scheme can keep transaction privacy from outside coercion. The transaction data on DBL can also be redacted or be forgotten. Our DBL construction uses the Chameleon Hash as the hash function to make a block be redactable since it is easy to find a collision with the trapdoor. So, the hash value of the redacted block is still the same to the original block and the original following block can still associated to the generated fake block. Even an attacker can collect multiple blockchains, it is impossible for the attacker to determine which blockchain is the original one. We also provide two scenarios and the corresponding algorithms of generating fake blocks. One is that every user is normal user and the other is that some users are compromised. We are currently implementing the DBL and hope that our construction can protect data privacy in the blockchain.

Keywords: Blockchain, Deniability, Chameleon Hash

† 通訊作者 (Corresponding author.)

壹、前言

區塊鏈這項技術對於現在的人們已經是毫不陌生，因為在這十年之內有大大小小的區塊鏈的應用被創造及開發，從比特幣、以太坊這些虛擬貨幣的使用到投票系統、供應鏈管理的發明等等，區塊鏈已經與這個科技日新月異的社會有著密不可分的關係。區塊鏈的架構與想法最早可以追溯到 1991 年由 Haber、Stuart 和 W. Scott Stornetta 所發表的論文[2]，雖然那個時候「區塊鏈」這個名字還沒有正式的出現，但當時這份論文已經提出了一個基於密碼學的鏈狀系統架構來防止資料的時間戳記遭受竄改。「區塊鏈」真正誕生並且聲名大噪的時機是當中本聰（Satoshi Nakamoto）在 2008 年發明了比特幣[3]—可以說是至今最具代表性及最廣為人知的虛擬貨幣—之後，這項技術才開始變成一門顯學，直到今天都還是有大量的學者與業者投入其中。區塊鏈簡單來說是一條基於密碼學的鏈狀架構，每一個區塊都儲存著需要被記錄的資訊，並且透過讓下一個區塊儲存上一個區塊的資訊的雜湊函數值來串連起這些區塊，又因為雜湊函數擁有不易找到相同函數值的特性，使得只要一修改前一個區塊內的資訊、下一個區塊所記載的雜湊函數值就無法與其呼應，讓區塊鏈有著廣為人知的「不可修改性」。

不可修改性本身是一個非常好的特性：例如我們可以藉由不可修改性來保證交易紀錄的正確與否，避免在沒有公正的第三方的情況下，有惡意的使用者利用修改交易紀錄來重複使用他的貨幣、或是將其他人的錢轉給自己；或是保障一些希望可以長時間保存的公開資料不受某些惡意的使用者的篡改，導致其他人在調閱或查看這些資料的時候無法確定哪一份複本才是正確的複本。但是不可修改性也有可能造成許多的不便及負擔：由於區塊鏈的不可修改性，我們沒有辦法去對已經生成的區塊做任何的新增、移除或修改，換句話說，我們沒有辦法去捨棄、否認我們曾經在區塊鏈上的所作所為，在「被遺忘權」逐漸受到世人所重視的現在，人們有權利要求有關於自己的負面、或是過時的個人資料受到遺忘，但在區塊鏈上卻沒有辦法做到被遺忘權的配合，因為同一個區塊內不一定所有資料的相關者都要求被遺忘，也不一定是先做成的區塊先要求被遺忘，導致「能保證區塊鏈內資料的完整性」這個優點在某個面向的討論上完全變成了缺點。

我們可以考慮一個情境：假設今天 Alice 想要將錢轉移到 Bob 的帳戶，但是雙方的帳戶可能在不同的地區、或是不同的國家，導致其實實際上金錢的流向在帳本中的紀錄可能會是 Alice—Cathy—Daniel—Bob，之後 Alice 將這些交易紀錄存在她私有的區塊鏈帳本中，可是突然有一個惡意的使用者出現並暴力脅迫 Alice 交出她的區塊鏈帳本，想要藉此來對 Cathy 進行攻擊，例如蒐集她的交易紀錄藉此判斷她身上有多少的資金可以運用、或是以她和某些人進行過交易來進行人身或名譽上的威脅，Alice 被迫必須照著惡意使用者的要求來做，她有沒有辦法動點手腳來保護中間的這些使用者？或者某段時

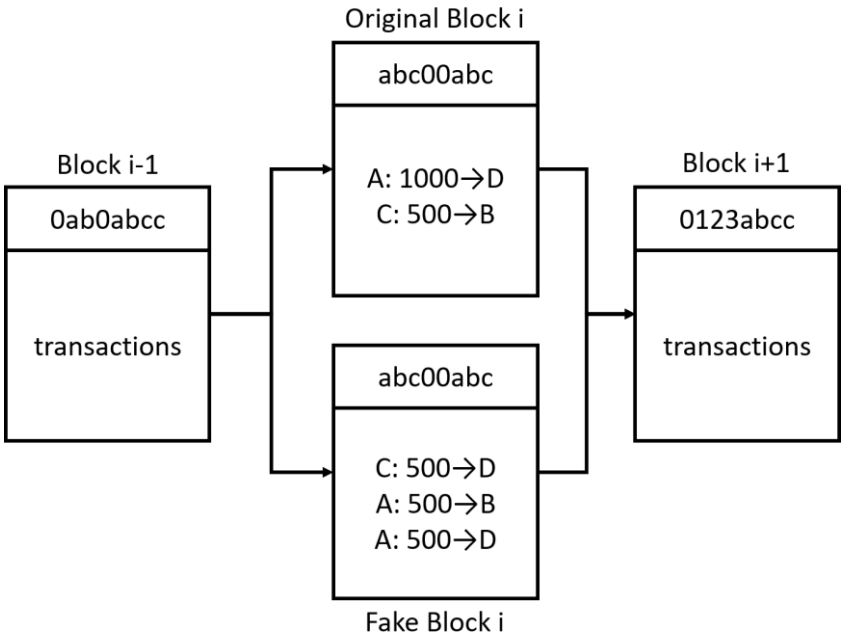

Figure 1: 可否認的區塊鏈帳本示意圖。我們的演算法可以生成具有相同雜湊函數值的區塊，所以不論是提供上下哪一個區塊當作第 i 個區塊都會是合法的區塊鏈。

間後，Cathy 因為某些私人的緣故，要求她當時協助這項交易的紀錄被遺忘，Alice 要如何在滿足 Cathy 的被遺忘權的前提下，同時的保證自己的區塊鏈帳本依舊能記載當時的其他資訊？Alice 或許可以嘗試著給予一份假的區塊鏈帳本，可是生成整條區塊鏈本身是非常花費時間成本的，很有可能會被惡意的使用者察覺；Alice 也或許可以嘗試著偷偷更改這一小部分和 Cathy 有關的交易紀錄，但是其他的使用者只要去比對每一個區塊的雜湊函數值就能發現這個區塊受過竄改。

為了解決類似的困境，我們提出了 DBL—可否認的區塊鏈帳本（Deniable Blockchain Ledger）這個架構，運用了 Ateniese 等人在 2017 年所提出的可編輯的區塊鏈架構[4]，讓使用者可以藉由生成擁有同樣雜湊函數值的交易紀錄區塊來否認原本正確的交易紀錄區塊的真實性，甚至這個演算法可以不斷的生成複數個區塊，讓所有保存這些交易紀錄的帳本複本都計載著不同的交易紀錄，來達到可否認性的應用，也能保障該使用者以外的其他相關使用者的權利。DBL 的主要架構在於利用 Chameleon Hash 這一項雜湊函數來進行區塊鏈中的雜湊函數運算並妥善地保留每一個區塊的暗門金鑰，當需要生成一個假的交易區塊來否認之前的區塊時，便生成符合該區塊的總收入支出、但中間參與交易的帳號是亂數生成及打散的交易紀錄，並透過暗門金鑰來做出該區塊的匹配字串，使得該區塊可以擁有與原本的區塊一樣的雜湊函數值，達到否認前一個區塊的效果。

貳、文獻探討

2.1 區塊鏈

區塊鏈是一種基於密碼學的鏈狀資料結構，主要是透過記錄前一個區塊的雜湊函數值來達到串接區塊的效果[1][3][9][10][11]。

區塊本身是一個三元組：$B=(s,x,ctr)$，其中 $s\in\{0,1\}^{\kappa}$ 代表的是前一個區塊的雜湊函數值，κ 為安全性參數；$x\in\{0,1\}^{*}$ 代表的是這個區塊所記載的資料或交易紀錄；$ctr\in\mathrm{N}$ 是一個在生成區塊時隨機產生的變數。驗證一個區塊是否為合法的區塊可以使用下列判斷式：

$$Ver_q^D(B)\equiv(H(ctr,G(s,x))<D)\wedge(ctr<q)=1$$

其中的 $D\in\mathrm{N}$ 是此區塊的難度等級、$q\in\mathrm{N}$ 是此區塊允許的雜湊計算次數的最大值，$H:\{0,1\}^{*}\to\{0,1\}^{\kappa}$ 和 $G:\{0,1\}^{*}\to\{0,1\}^{\kappa}$ 皆為抗碰撞性的雜湊函數。

區塊鏈即為鏈狀的區塊、或是區塊的序列，我們將區塊鏈 C 最尾端的區塊稱為區塊鏈的頭，符號記為 $Head(C)$，一條 $Head(C)=(s,x,ctr)$ 的區塊鏈 C 可以藉由串接一個新的區塊 $B'=(s',x',ctr')$ 來做成一條較長的區塊鏈 $C'=C\|B'$，但是條件如下：

$$s'=H(ctr,G(s,x))$$

區塊鏈根據是否中心化來分成公有鏈與私有鏈兩種：公有鏈屬於非中心化的區塊鏈，大部分的虛擬貨幣都屬於公有鏈的體系，公有鏈中沒有一個中央的管理者、所有使用者都是等價的使用者，所以公有鏈會依靠共識系統或是最長鏈規則來判斷使用者所儲存的區塊鏈是否和其他使用者儲存的一致，也避免有惡意的使用者去修改鏈的內容並試圖欺騙其他使用者；私有鏈則為中心化的區塊鏈，通常存在於公司內部的私有帳本或組織內部的私有紀錄中，私有鏈最主要的功用即為儲存該組織所需要記錄的資料，所以會有一個權限最高的管理者，以該管理者所擁有的鏈作為依據，其他的使用者可以向管理者要求保存鏈的複本。本文中所提出的可否認的區塊鏈帳本主要會在私有鏈的方面進行討論，我們會有一個帳本的擁有者來做為整條鏈的管理者並儲存有關於鏈的相關資訊，並擁有提供其他使用者區塊鏈的複本的權限。

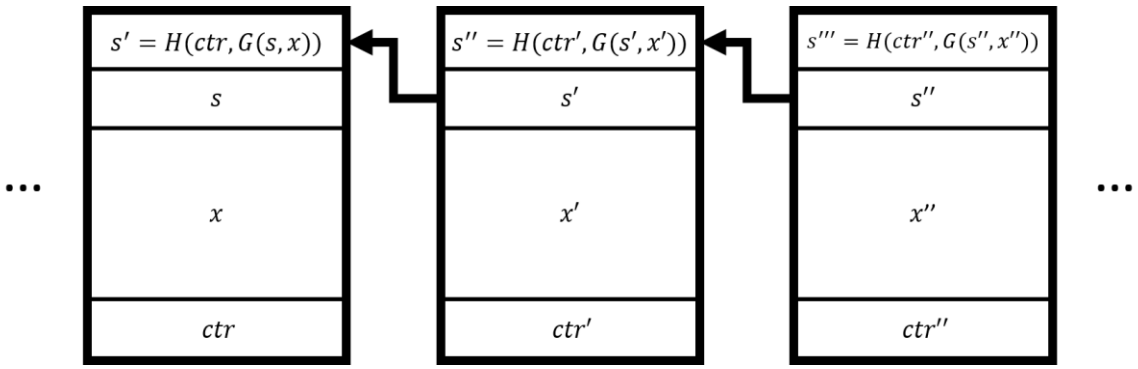

Figure 2: 區塊鏈示意圖

2.2 Chameleon Hash

Chameleon Hash 是一種抗碰撞性的雜湊函數，但是其特點在於該函數在計算雜湊函數值之前會先生成一把公開的雜湊金鑰和一把私密的暗門金鑰：雜湊金鑰用於計算雜湊函數值和雜湊函數值與原訊息字串的驗證，而暗門金鑰則用於計算雜湊函數值的碰撞，也就是說，在沒有暗門金鑰資訊的情況下，Chameleon Hash 是一種抗碰撞性的雜湊函數，但是只要有了暗門金鑰，便可以在有限的時間內取得雜湊函數值的碰撞。

Chameleon Hash 由以下四個演算法所組成：

1. $(hk, tk) \leftarrow CHGen(1^{\kappa})$：此演算法的輸入為安全性參數 κ，之後輸出 hk 作為公開的雜湊金鑰、tk 作為私密的暗門金鑰。
2. $(h, \xi) \leftarrow CH(hk, m)$：此為計算雜湊函數值的演算法，透過輸入雜湊金鑰和訊息字串來計算出一組雜湊函數值 h 和驗證字串 ξ。
3. $d \leftarrow CHVer(hk, m, (h, \xi))$：是一個驗證雜湊函數值與訊息字串是否吻合的演算法，若 $CH(hk, m) = (h, \xi)$ 則輸出 1、反之則輸出 0。
4. $\xi' \leftarrow CHCol(tk, (h, m, \xi), m')$：此為計算雜湊函數值的碰撞的演算法，倘若我們希望讓 m' 擁有和 m 一樣的雜湊函數值，我們便利用此演算法去找到一個 ξ' 使得 $CHVer(hk, m, (h, \xi)) = CHVer(hk, m', (h, \xi')) = 1$。

此外，Chameleon Hash 有三個重要的性質：

1. **防碰撞性**：沒有任何足夠快速且有效的演算法能夠在使用了公開的雜湊金鑰 hk 作為輸入後，找到兩組字串組 $(m_1, \xi_1), (m_2, \xi_2), m_1 \neq m_2$ 使得 $CHVer(hk, m_1, (h, \xi_1)) = CHVer(hk, m_2, (h, \xi_2)) = 1$

2. **可偽造性**：存在一個足夠快速且有效的演算法能夠在輸入暗門金鑰tk、任意一組字串組(m_1,ξ_1)以及一個任意的訊息字串m_2之後，找到一個字串ξ_2使得$CHVer(hk,m_1,(h,\xi_1))=CHVer(hk,m_2,(h,\xi_2))=1$
3. **一致性**：所有訊息字串m在 Chameleon Hash 中都擁有一樣的機率，也就是說，就算(h,ξ)是公開的，知道了這組資訊也沒有辦法帶來任何跟m有關的訊息。

到今天為止，有許多的學者利用了各式各樣的難題來建構出 Chameleon Hash 並實際的應用，例如 Krawczyk 和 Rabin 在 2000 年提出了建立在離散對數難題上的 Chameleon Hash 架構[5]、Zhang 等人於 2003 年做出了建構在計算的 Diffie-Hellman 難題上且能夠進行身分認證的 Chameleon Hash[6]、Ateniese 等人更在 2004 年將 Chameleon Hash 架構在 RSA 演算法上[8]。而 Chameleon Hash 也有許多相關的應用，如 Ateniese 等人於 2004 年利用其架構作出了包含身分認證架構的簽章演算法[7]、Mohassel 等人在 2010 年發表了將任何 Chameleon Hash 轉換成不可偽造的一次性簽章[12]，其中最著名的或許是 Ateniese 等人在 2005 年所發表的 Sanitizable 簽章演算法[14]，該演算法允許可信任的第三方在更動了資料內容後自行修正整份資料的簽章而不用再透過原本簽章者之手，使得便利性有著大幅度的提高，在軍方、醫學、多媒體以及路由上都有著貢獻。

2.3 可修改的區塊鏈

可修改的區塊鏈是由 Ateniese 等人在 2017 年所發表的新形態的區塊鏈架構[4]，該架構顧名思義是一條沒有不可修改性的區塊鏈，擁有授權的使用者或是區塊鏈的管理者能夠修改區塊鏈中某個單一區塊內部的資料且保持其他區塊的資料不做更動，不需要因為修改了單一區塊而導致該區塊的雜湊函數值發生變化、進而必須更動該單一區塊後的所有區塊。讓區塊鏈內的單一區塊能夠進行修改而不影響雜湊函數值的原因即是因為該區塊在計算雜湊函數值時使用的是 Chameleon Hash，所以在更動了單一區塊的內容之後，只要利用$CHCol$計算出新的資料內容的雜湊函數值組(h,ξ')並將原本結構內的ξ替換成ξ'，即可在不更動到後面區塊的前提下進行單一區塊內資料的修改。

可修改的區塊鏈內的區塊結構$B=(s,x,ctr,(h,\xi))$，其中$(h,\xi)=CH(hk,(s,x))$為一組雜湊函數值組，驗證該區塊是否為合法區塊的判斷式則變為：

$$Ver_q^D(B)\equiv(H(ctr,h)<D)\wedge(CHVer(hk,(s,x),(h,\xi))\wedge(ctr<q)=1$$

且若一條$Head(C)=(s,x,ctr,(h,\xi))$的可修改的區塊鏈$C$可以串接一個新的區塊$B'=(s',x',ctr',(h',\xi'))$形成新的區塊鏈$C'=C\,||\,B'$，則該區塊$B'$須滿足的條件式為$s'=H(ctr,h)$。

而修改區塊的演算法如下：

輸入：長度為 n 的區塊鏈 C 、需要被修改的區塊集合 $\mathrm{I} \subseteq [n]$ 、修改後的值的集合 $\{x_i'\}_{i \in \mathrm{I}}$ 、暗門金鑰 tk
輸出：修改過後長度為 n 的區塊鏈 C'

$C' \leftarrow C$;
Parse the chain C' as $(B_1, B_2, ..., B_n)$;
for $i = 1$ to n do
  if $(i \in \mathrm{I})$ then
    Parse the i-th block of C' as
    $B_i \equiv (s_i, x_i, ctr_i, (h_i, \xi_i))$;
    $\xi_i' \leftarrow CHCol(tk, (h_i, s_i \parallel x_i, \xi_i), (s_i \parallel x_i'))$;
    $B_i' \equiv (s_i, x_i', ctr_i, (h_i, \xi_i'))$;
  end
end
return C'

Figure 3: 修改區塊的演算法

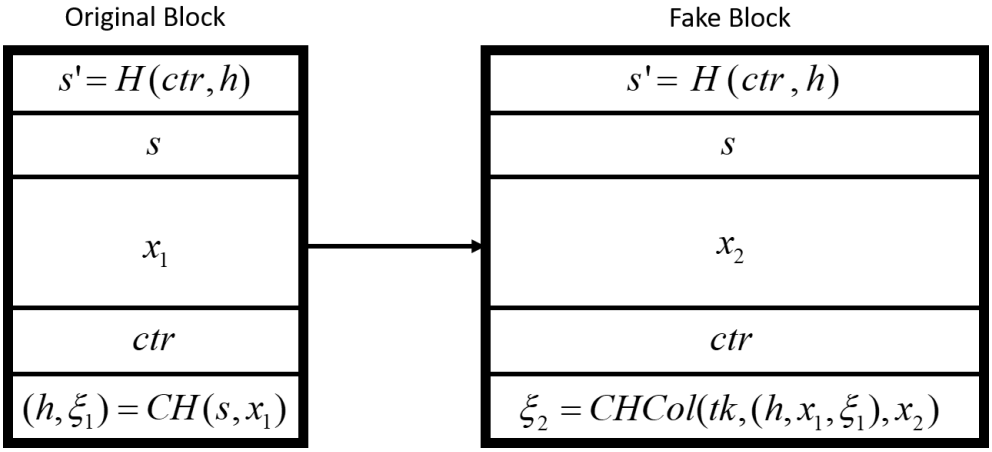

Figure 4: 可修改的區塊鏈示意圖，使用 CHCol 演算法製造雜湊碰撞，讓修改了交易紀錄後的區塊能保持一樣的雜湊函數值

另外，2019 年 Derler 等人也提出了一個基於 Chameleon Hash 的可修改式區塊鏈架構[13]，相較於 Ateniese 等人將 Chameleon Hash 使用在整個交易紀錄之上，Derler 等人則是用 Merkle Hash Tree 紀錄交易的資料並將雜湊碰撞用在建立 Merkle Hash Tree 上，讓每一筆交易紀錄都可以在這個樹狀結構裡作出雜湊碰撞，這樣在作區塊的修改時只要修改單一的交易紀錄而不用重新計算整個區塊的雜湊碰撞，大大的降低了修改區塊所需花費的時間並提升了效率。

參、方法

在這個章節我們會介紹我們的區塊鏈架構，並解釋在我們的架構之下，如何去保障除了兩個端點帳號以外的中間所有帳號，以及如何在保護中間所有交易帳號的同時不去影響區塊鏈的雜湊值檢驗，使其依舊為合法的區塊鏈。

我們最主要的想法是透過 Chameleon Hash 在擁有暗門金鑰的狀態下，製造數個不同的交易紀錄去生成不同的區塊，但是這些區塊雖然在交易內容和參數上不盡相同，使用了 Chameleon Hash 的雜湊碰撞計算後卻都有著一樣的雜湊函數值，而當有其他的使用者來要求獲得這條區塊鏈的複本時，我們便在有需要被否認的區塊位置上使用我們的演算法來做出一個擁有假的交易紀錄、但是同時擁有和真實區塊一樣的雜湊函數值的區塊，這樣相同的雜湊函數值可以讓下一個區塊紀錄下來並完成合法的區塊鏈的串接，讓其他的使用者無法透過檢驗區塊鏈是否合法的方式來判斷哪一個區塊的交易紀錄才是真實的交易紀錄，因為每一個人拿到的區塊鏈複本，都會是一條合法的區塊鏈。

3.1 前提假設

我們假設我們所使用的區塊鏈為私有鏈，私有鏈的擁有者為一個安全且可信任的使用者，其他的使用者可以向擁有者要求保存該私有鏈的複本以及要求所有公開的參數，而因為修改區塊鏈所需要的暗門金鑰只有擁有者可以知曉，所以所有修改區塊鏈的演算法僅能由擁有者實行。

3.2 基本架構與演算法

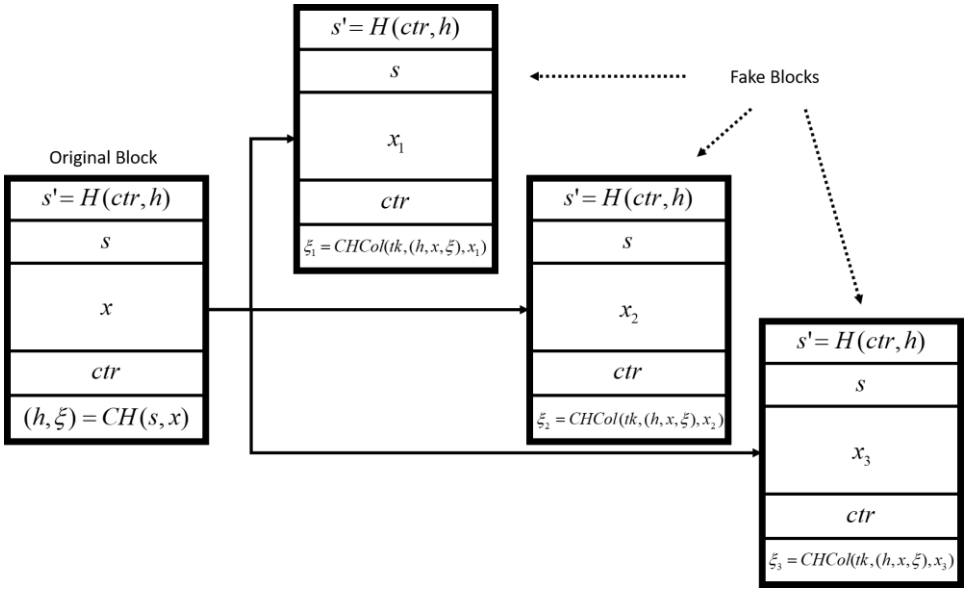

Figure 5: 我們可以做出許多假的交易紀錄區塊，各自配合各自的驗證字串使得這些假區塊都和左邊的真區塊有著一樣的雜湊函數值，讓我們在給予帳本複本的時候可以給予包含假區塊的複本，拿到複本的人就算去檢驗也只能確認這是一條合法的區塊鏈、沒有證據說明我們提供的複本有包含假的區塊。

我們先考慮所有與交易紀錄相關的使用者皆為一般使用者的情況，也就是說在區塊鏈中有過交易紀錄的使用者都不是惡意的使用者、或是惡意的使用者的同夥。在這個情況下，我們會通過幾個步驟來製作可否認式的區塊鏈帳本，以及製作假的區塊紀錄來保障該區塊的可否認性以及被遺忘權：我們的區塊鏈架構為 $B=(s,x,ctr,(h,\xi))$，$s\in\{0,1\}^{\kappa}$ 為前一個區塊的雜湊函數值，若該區塊為最初始的區塊則設 $s=0^{\kappa}$，$x\in\{0,1\}^{*}$ 為該區塊內所記載的所有交易紀錄，$ctr\in\mathrm{N}$ 做為一個生成區塊時隨機產生的亂數，$(h,\xi)=CH(hk,(s,x))$ 為該區塊的 Chameleon Hash 雜湊函數值組。在生成一個新的區塊時，擁有者會先啟用 $(hk,tk)\leftarrow CHGen(1^{\kappa})$ 來生成一組 Chameleon Hash 的金鑰，hk 是公開的雜湊金鑰，用來讓所有使用者進行雜湊函數值的生成和驗證，tk 則是私密的暗門金鑰，用來讓區塊鏈的擁有者進行雜湊碰撞的計算，接著擁有者會去目前區塊鏈的尾端取得該區塊的雜湊函數值 h，並以具有碰撞抗性的雜湊函數 H（例如 Sha256）計算得 $s=H(ctr,h)$ 作為接續前一個區塊的橋樑，並寫下要儲存在該區塊的交易紀錄 $x\in\{0,1\}^{*}$，再來透過一個不可預測的隨機值生成器生成一個 $ctr\in\mathrm{N}$，最後再使用 $(h,\xi)=CH(hk,(s,x))$ 記錄下這個區塊的雜湊函數值組，完成這一個區塊的建立。此外，考量到 Chameleon Hash 的安全性，我們每一個區塊都會重新建立一組 Chameleon Hash 的雜湊金鑰以及暗門金鑰並由管理者記錄下來，當有區塊需要建立假的區塊複本

時，管理者便去查詢該區塊建立時的金鑰數值並使用即可，這樣子不重複使用金鑰雖然會增加些許的儲存空間，但是可以避免同一把金鑰因為使用太久而被破解。

輸入：目標區塊 $B=(s,x,ctr,(h,\xi))$ 、該區塊對應的暗門金鑰 tk 、所有使用者的集合 U 、隨機交易使用者數量變數 p
輸出：假區塊 $B'=(s,x',ctr,(h,\xi'))$

Parse $x = x_1 \| x_2 \| ... \| x_{|x|}$;
for $k=1$ to $|x|$ do
    Parse $x_k = (u_E, n, u_I)$ %代表使用者 u_E 轉移 n 塊錢給使用者 u_I
    for $l=1$ to p do
        $u_R \xleftarrow{R} U, u_R \neq u_E$、$u_I$;
        $x_k' \leftarrow (u_E, n, u_R)$;
        $u_E \leftarrow u_R$;
    end
    $x_k' \leftarrow (u_R, n, u_I)$;
end
$x' = x_1' \| x_2' \| ... \| x_{|x|}'$;
$\xi' \leftarrow CHCol(tk, (h, (s \| x)), \xi), (s \| x'))$;
$B' = (s, x', ctr, (h, \xi'))$
return B';

Figure 6: DBL 演算法

當有一般使用者或是惡意的使用者來要求區塊鏈擁有者交出區塊鏈的複本時，擁有者可以利用下面這個 DBL 演算法生成一個假的交易紀錄區塊，透過先前所提到的 Chameleon Hash 的特性以及我們所使用的區塊鏈架構，我們將此虛偽的交易紀錄區塊去取代原本的區塊之後，其他的使用者並沒有辦法去判斷這個區塊是否被動過手腳，因為所有的雜湊函數值都是一模一樣的、整條區塊鏈依舊是合法的區塊鏈，藉此達到我們架構的「可否認性」。

我們的 DBL 演算法主要的精神是這樣的：我們會先將交易紀錄分解出支出使用者、收入使用者以及轉移的金額三個參數，在生成假交易紀錄的部分，我們隨機從所有使用者的集合中內挑出一個不是收入或支出使用者的使用者當作中間人，產生一筆支出使用者將金額轉移給中間人以及多筆中間人將金額轉移給中間人的交易紀錄，最

後一位中間人將金額轉移給收入使用者後便把所有假交易紀錄整理並透過*CHCol*演算法來製造ξ'，使得這些交易紀錄進行雜湊函數運算後依舊能和原本的雜湊函數值相同，我們便創造出了一個合法的假交易區塊，其他使用者並沒有辦法證明這個區塊的真偽。

3.3 延伸情況及其演算法

我們接下來會討論一個延伸的情況：如果在區塊鏈帳本中所紀錄到的某些使用者是惡意的使用者或其同夥。由於在這個狀況下，這些惡意的使用者或其同夥（以下簡稱同夥使用者）擁有與自己相關的交易紀錄資料，所以如果使用一般情況的演算法，在隨機生成中間交易過程的情況下便會產生破綻，讓同夥使用者可以以自己的交易紀錄來證明這個假的區塊是假的，導致原本的區塊失去了可否認性。為了解決這個問題，假設區塊鏈的擁有者可以知道某些使用者為同夥使用者，令U為所有使用者的集合、且這些同夥使用者的集合為$U_M \in U$，擁有者可以利用 DBL-Extend 演算法來避開這些同夥使用者，僅更動不屬於同夥使用者的交易紀錄，保障原本區塊的可否認性，在這個延伸演算法中，我們同時也建構了比較複雜的假交易紀錄生成機制，以防太過單純的交易紀錄會讓其他使用者起疑心。

在 DBL-Extend 演算法中，我們主要的目的是避開和同夥使用者相關的交易紀錄，並且隨機打散剩下的所有交易紀錄，所以首先我們一樣將交易紀錄分解出支出使用者、收入使用者以及轉移的金額三個參數，並先清空x'裡面的值，接著我們便先對所有交易紀錄去進行判定，如果支出或收入有其中一方屬於同夥使用者的集合，我們就直接將這筆交易紀錄儲存在x'中，並且在這兩個使用者的交易前後的帳戶金額值內刪掉這筆轉移的金額，避免之後隨機產生交易紀錄時發生影響。處理完與同夥使用者有關的交易後，我們便開始隨機產生交易紀錄，我們的方法是去依序判斷每一個使用者在交易前後的帳戶金額值的增減，如果這名使用者是同夥使用者就跳過，若為一般的使用者則判斷他在交易前後的總和是增加還是減少，若是減少的話我們便進入一個迴圈，迴圈內會先隨機的在所有非同夥使用者中找出交易前後的總和是增加的使用者，並讓他們進行金錢的轉移並記錄在x'內，轉移後還有餘額則繼續迴圈、剛好打平或是轉移的錢不夠則跳出迴圈並找尋下一個交易前後總計是減少的使用者，當我們檢查完所有的非同夥使用者後，這些人的收支便會達到平衡，我們就可以把這些假的交易紀錄拿去進行雜湊碰撞，透過*CHCol*的運算得到ξ'之後，即可做出一個擁有與原本區塊相同函數值的假區塊複本。

輸入：目標區塊 $B=(s,x,ctr,(h,\xi))$ 、該區塊對應的暗門金鑰 tk 、所有使用者的集合 U 、同夥使用者的集合 U_M 、該區塊交易發生前各使用者帳戶內的金額值 $P[|U|]$ 、交易發生後的金額值 $N[|U|]$
輸出：假區塊 $B'=(s,x',ctr,(h,\xi'))$

Parse $x = x_1 \| x_2 \| ... \| x_{|x|}$; $x' \leftarrow \perp$;
for $k=1$ to [illegible] do
    Parse [illegible] %代表使用者 n 轉移 n 塊錢給使用者 u_I
    if ($u_E \in U_M$ or $u_I \in U_M$) then
        $x' \leftarrow x' \| x_k$;
        $P[u_E] \leftarrow P[u_E] - n$;
        $N[u_I] \leftarrow N[u_I] - n$;
end
for $j=1$ to $|U|$ do
    if $u_j \in U_M$ then
        skip;
    else
        if $N[j] < P[j]$ then
            $temp \leftarrow P[j] - N[j]$;
            while ($temp \neq 0$) do
                $u \xleftarrow{R} U - U_M$, $u \neq j$;
                if ($N[u] - P[u] \geq temp$) then
                    $P[u] \leftarrow P[u] + temp$;
                    $x' \leftarrow x' \| (u_j, temp, u)$;
                    $temp \leftarrow 0$;
                else if ($0 < N[u] - P[u] \leq temp$)
                    $P[u] \leftarrow N[u]$;
                    $x' \leftarrow x' \| (u_j, N[u] - P[u], u)$
                    $temp \leftarrow temp - (N[u] - P[u])$;
            end
end
[illegible] ;
$B'=(s,x',ctr,(h,\xi'))$
return B';

Figure 7: DBL-Extend 演算法

肆、討論

4.1 帳本複本的給予規則

在 DBL 這條可否認式的區塊鏈上的所有使用者都有權利來向這條鏈的管理者要求儲存 DBL 的複本，所以我們在基於安全以及隱私的考量下會可以對於複本的給予建立一些規則。在給予複本時，我們可以利用 DBL-Extend 演算法來將要求儲存複本的使用者視為同夥使用者U_M的唯一一個集合元素，讓演算法在避開這名使用者的交易紀錄的同時，產生一連串的假交易紀錄去取代與這名使用者無關的交易紀錄。這個舉動並不是為了預防這名使用者成為對我們的 DBL 或是對其他的使用者產生威脅的人，而是為了保障其他使用者的隱私權，我們知道在公開的區塊鏈上所有的交易紀錄都是記錄在區塊內的，使用者並沒有辦法去隱藏這些與個人相關的資訊，這對於現在注重個人隱私權的社會而言著實是一項缺點，但如果我們使用 DBL-Extend 演算法來給予複本，我們可以讓該名使用者所拿到的複本中只有與自己有關的交易紀錄是正確無誤的、其他的交易紀錄皆是隨機產生的，這樣除了可以讓使用者確實保存與自己相關的交易紀錄，也可以防止其他使用者的交易紀錄被公開，保障 DBL 上的所有使用者的隱私及安全。

4.2 Chameleon Hash 的暗門金鑰洩漏問題

實際上 Chameleon Hash 的防碰撞性在某一個特定狀況下會有洩漏暗門金鑰的風險，我們以下用建立在離散對數難題上的 Chameleon Hash 來做舉例[8]。建立在離散對數南堤上的 Chameleon Hash 會先取兩個質數p,q且滿足$p=kq+1$，再取g是Z_p中一個q階元素，接著隨機產生一把暗門金鑰$x\in Z_q$，雜湊金鑰則為$y=g^x \bmod p$，對於一個字串m以及一個隨機配對字串r，$CH(m,r)=g^m y^r \bmod p$即是這串字串的雜湊函數值，而對於另一個字串m'，製造出令其產生雜湊碰撞的配對字串r'的演算法為$Forge(m,r,m')=r'=((m-m'+xr)/x) \bmod q$。在觀察這個演算法之後其實我們可以發現，要是某人能夠同時擁有兩組發生雜湊碰撞的字串$(m,r),(m',r')$，他便有機會可以找到暗門金鑰x，方法如下：

$$g^m y^r \bmod p = g^{m'} y^{r'} \bmod p$$
$$g^{m+xr} = k(g^{m'+xr'})$$
$$m+xr = \log_g k + m'+xr'$$
$$x = (\log_g k + m'-m)/(r-r')$$

所以如果某一位使用者手上握有兩個不同的 DBL 複本，他便有機會透過上方的運算式去推得私密的暗門金鑰x。

暗門金鑰的洩漏對我們的 DBL 是否會產生安全上的問題？這個答案是否定的。的確，如果今天有某位使用者和其他夥伴串通好，分別從 DBL 的管理者方拿到了兩份不一樣的 DBL 複本，他們有機會透過運算來得到私密的暗門金鑰，但是得到了這份金鑰的唯一一個用處便是他們可以自行製作假的交易區塊，而事實上使用者自行建立假的交易區塊對整體區塊鏈是毫無影響的，因為我們的 DBL 屬於私有鏈的範疇，使用者想要獲得複本時必須向管理者去要求儲存複本，所以其他使用者自行建立複本並不會影響到複本的給予，而就算能夠自行建立複本，原始的交易紀錄依舊只有管理者能夠擁有，暗門金鑰的洩漏沒辦法讓惡意的使用者去回推原本的交易紀錄，因為這些交易紀錄都是隨機產生的、與原來的交易紀錄除了總金額的增減相同以外沒有任何關係，所以雖然 Chameleon Hash 擁有這個暗門金鑰洩漏的問題，我們的 DBL 對於使用者而言仍然是安全無虞的。

伍、結論

本文提供了一個名為 DBL─可否認的區塊鏈帳本的區塊鏈架構，這個架構賦予了區塊鏈型態的帳本可否認性，讓區塊鏈帳本的擁有者可以自由地更改區塊鏈帳本中所記載的交易紀錄資料，以防惡意的使用者對擁有者進行暴力脅迫。擁有者透過 Chameleon Hash 來製造雜湊函數值的碰撞來生成不同的假交易紀錄區塊，當有使用者或惡意的使用者要求擁有者交出區塊鏈的複本時，擁有者便可以交出一條含有假交易區塊的區塊鏈，但是由於假交易區塊會產生和原交易區塊一樣的雜湊函數值，使得其他人在檢驗這條區塊鏈時都只能檢驗出它是一條合法的區塊鏈，沒有辦法提出證據證明擁有者造假。未來我們會試著實際的做出可否認的區塊鏈帳本，並修改成能在公有鏈上運行的版本使其能夠套用在如比特幣或是以太坊之類的現有區塊鏈架構中，測試它耗費時間和耗費空間的表現，也期望這個架構能夠在現實中獲得更多其他的應用。

[誌謝]

本文完成感謝科技部計畫 MOST 107-2218-E-003-002-MY3 的支持。

參考文獻

[1] Pilkington, Marc. "11 Blockchain technology: principles and applications." Research handbook on digital transformations 225 (2016).

[2] Haber, Stuart, and W. Scott Stornetta. "How to time-stamp a digital document." Conference on the Theory and Application of Cryptography. Springer, Berlin, Heidelberg, 1990.

[3] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008).

[4] Ateniese, Giuseppe, et al. "Redactable blockchain–or–rewriting history in bitcoin and friends." 2017 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 2017.

[5] Krawczyk, Hugo Mario, and Tal D. Rabin. "Chameleon hashing and signatures." U.S. Patent No. 6,108,783. 22 Aug. 2000.

[6] Zhang, Fangguo, Reihaneh Safavi-Naini, and Willy Susilo. "ID-Based Chameleon Hashes from Bilinear Pairings." IACR Cryptology ePrint Archive 2003 (2003): 208.

[7] Ateniese, Giuseppe, and Breno de Medeiros. "Identity-based chameleon hash and applications." International Conference on Financial Cryptography. Springer, Berlin, Heidelberg, 2004.

[8] Ateniese, Giuseppe, and Breno de Medeiros. "On the key exposure problem in chameleon hashes." International Conference on Security in Communication Networks. Springer, Berlin, Heidelberg, 2004.

[9] Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger." Ethereum project yellow paper 151.2014 (2014): 1-32.

[10] Swan, Melanie. Blockchain: Blueprint for a new economy. " O'Reilly Media, Inc.", 2015.

[11] Bayer, Dave, Stuart Haber, and W. Scott Stornetta. "Improving the Efficiency and Reliability of Digital Time-Stamping Sequences." (1992): 329334.

[12] Mohassel, Payman. "One-time signatures and chameleon hash functions." International Workshop on Selected Areas in Cryptography. Springer, Berlin, Heidelberg, 2010.

[13] Derler, David, et al. "Fine-Grained and Controlled Rewriting in Blockchains: Chameleon-Hashing Gone Attribute-Based." IACR Cryptology ePrint Archive 2019 (2019): 406.

[14] Ateniese, Giuseppe, et al. "Sanitizable signatures." European Symposium on Research in Computer Security. Springer, Berlin, Heidelberg, 2005.