適用電子商務環境之金鑰恢復與託管機制研究

曹偉駿

大葉大學資訊管理學系

wjtsaur@mail.dyu.edu.tw

林幸君

大葉大學資訊管理學系

shingjiun@yahoo.com.tw

摘要

在電子商務環境中，一個安全的交易環境是不可或缺的，而安全的密碼系統正提供我們一個很好的解決方案。本研究所要探討的課題即是密碼系統中的金鑰恢復與金鑰託管機制。由於金鑰恢復與金鑰託管機制之運作需要繁複的金鑰運算與訊息傳遞。而本研究提出之金鑰恢復與託管機制，利用橢圓曲線密碼系統中金鑰長度較短與計算速度快之優點，以及自我認證公開金鑰密碼系統中有效率的驗證公鑰正確性之優點，讓此機制不必依靠系統中心就能達到身份確認以及提高效率性。另外我們在電子現金系統中加入託管的功能，設計出託管電子現金系統，避免電子現金完全匿名可能產生之犯罪行為，我們所設計的託管電子現金系統除了具有防範網路犯罪之功能(如洗錢、勒索)，並且能迅速地找出犯罪者，還給受害者一個公道。

關鍵詞：橢圓曲線密碼系統、自我認證公開金鑰密碼系統、金鑰恢復機制、金鑰託管機制、電子現金系統。

1. 前言

在電子商務環境中，金鑰恢復與託管機制提供的金鑰管理技術讓使用者能安心的使用密碼系統。

金鑰託管機制能確保使用者通信之秘密性，並防止不當使用者的網路犯罪行為[2]；金鑰恢復機制則能協助使用者重新取回遺失或毀損之解密金鑰，避免商業上的損失[8]。此外，在電子商務環境中，電子付款機制因其便利的特性也漸漸被發展，其中的電子現金系統較為大眾所接受[13]。但電子現金系統中的匿名特性卻可能會造成網路犯罪的產生，因此探討如何在必要時廢止電子現金的匿名性是很有意義的。

2. 金鑰恢復與託管機制

傳統的金鑰恢復與託管機制，存在有金鑰產生、訊息傳送及金鑰儲存等階段，在傳統機制中公鑰的認證常需要第三公正單位的協助，以及儲存公鑰憑證，如此必須花費較大的計算時間與儲存空間[6, 7, 8, 10, 12]。本研究目的在於設計出一個既安全又有效率的金鑰恢復與託管機制，減少金鑰恢復與託管機制之運算量及通訊量，並且能有效率的驗證公鑰之正確性，讓金鑰恢復與託管機制能更有效率的提昇電子商務環境之安全。我們利用陳[1]提出的基於橢圓曲線密碼系統之自我認證公開金鑰密碼系統與 Nechvatal[7]基於公開金鑰之金鑰託管系統之觀念來設計我們提出的機制。以下說明此機制之組成要素：

(1)系統中心(System Authority, SA)：協助使用者、金鑰託管單位產生其公私鑰，並且回傳公鑰與公鑰證明。在合法授權情況下，經由金鑰託管單位的協助重新建構恢復使用者的交談金鑰(session key, SK)，進而解密加密的訊息。

(2)金鑰託管單位(Key Escrow Agent, KEA)：協助使用者產生 *LEAF*(Law Enforcement Access Field)，並於合法授權情況下，協助系統中心重新建構恢復使用者的 SK。

(3)子金鑰託管單位($KEA1_i$, $KEA2_j$)：協助金鑰託管單位託管其金鑰，並在必要時，提供足夠之子金鑰給系統中心。

2.1 系統建置階段

系統中心的初始化，用以建立公開及秘密參數。由 SA 計算以下參數：

(1)p：在一般應用中 p 為一個奇數質數或者是 2 的乘幕次方，長度為 160 位元。

(2)F_p 場中的橢圓曲線 E：

E：$y^2=x^3+ax+b \pmod p$，其中 a、$b\in F_p$，$4a^3+27b^2\neq 0 \pmod p$，且在 E 上的所有點$(x, y)\in F_p$，$E(F_p)$集合中包含一個點無窮遠點 O。

(3)B：在 $E(F_p)$中序為 q 的基點，q 為一個 160 位元的大質數，其中$\#E(F_p)$為在 E 之中與 F_p 相關的點的總數，且可被 n 除盡。

(4)P_{SA}：系統中心的公鑰，其中 $P_{SA}=s_{SA}\cdot B \pmod p$(“ · ”表示為數值與橢圓曲線點的乘法運

算)。
(5)s_{SA}：系統中心的私鑰，且 $s_{SA}\in[2, q-2]$。
(6)$h(\ )$：單向雜湊函數(One-way hash function)，允許可變的長度輸入然後產生一個固定長度輸出值 j，$j\in[2, q-2]$且為 160 位元。
(7)P_i：使用者 U_i 的公鑰。
(8)s_i：使用者 U_i 的私鑰。
(9)(P_{KEA1}, S_{KEA1})：KEA1 的公私鑰。
(10)(P_{KEA2}, S_{KEA2})：KEA2 的公私鑰。
(11)w_i：使用者的公鑰證明(Witness)。
(12)$X(P)$：輸出點 P 在 X 座標軸上的值。
(13)I_i：使用者 U_i 的身份資訊。
當 SA 建立這些參數後，將$\{E, B, p, q, P_{SA}, h(\ )\}$公開，保留 s_{SA} 當成其密鑰。

2.2 使用者駐冊階段

使用者 U_i 執行以下步驟向 SA 註冊以取得合法公鑰及證明，並計算其密鑰。
1.使用者 U_i 隨機選擇一個亂數 x_i，且計算 $V_i=h(x_i, I_i)\cdot B \pmod p$，並將自己的身分 I_i 與 V_i 傳送給 SA。
2.SA 選擇一個亂數 k_i，且計算使用者 U_i 的公鑰為 $P_i=V_i+(k_i-h(I_i))\cdot B \pmod p$與公鑰證明 $w_i=k_i+s_{SA}\cdot(X(P_i)+h(I_i)) \pmod q$，並傳送 P_i 與 w_i 給使用者 U_i。
3.使用者 U_i 計算自己的私鑰 $s_i=w_i+h(x_i, I_i) \pmod q$，並驗證公鑰的有效性是否成立：
$$s_i\cdot B=P_i+h(I_i)\cdot B+(X(P_i)+h(I_i))\cdot P_{SA} \pmod p \quad (1)$$
若驗證成功，則使用者 U_i 的私鑰為 s_i，公鑰為 P_i。
定理 1：使用者可於註冊階段驗證從 SA 取得的公鑰 P_i 之有效性。
證明：
$s_i\cdot B=(w_i+h(x_i, I_i))\cdot B \pmod p=w_i\cdot B+h(x_i, I_i)\cdot B \pmod p=(k_i+s_{SA}\cdot(X(P_i)+h(I_i)))\cdot B+h(x_i, I_i)\cdot B \pmod p=(k_i+h(x_i, I_i))\cdot B+(X(P_i)+h(I_i))\cdot P_{SA} \pmod p=k_i\cdot B+V_i+(X(P_i)+h(I_i))\cdot P_{SA} \pmod p=P_i+h(I_i)\cdot B+(X(P_i)+h(I_i))\cdot P_{SA} \pmod p$
Q.E.D.
KEA1 與 KEA2 註冊方式與上述相同。

2.3 金鑰託管階段

此階段使用 Pedersen[9]提出之可驗證式秘密分享機制將 KEA1 與 KEA2 之秘密金鑰 S_{KEA1} 與 S_{KEA2} 分別分享到 $KEA1_i$ 與 $KEA2_j$ 中，達成金鑰託管的目的。KEA1 運作金鑰託管過程如下：
(1)KEA1 選擇一個 $t-1$ 次方的多項式 $f(x)=a_0+a_1x+\cdots+a_{t-1}x^{t-1}$，令 $a_0=S_{KEA1}=f(0)$，每個 $KEA1_i$ 都有一個獨一無二的身份 I_{KEA1_i}，計算 $S_{KEA1_i}=f(I_{KEA1_i})$，$i=1\sim n$，並且秘密將 S_{KEA1_i} 分別送給 $KEA1_i$。
(2)KEA1 將驗證序列 $K=(P_{KEA1_1}=S_{KEA1_1}\cdot B \pmod p, P_{KEA1_2}, \cdots, P_{KEA1_n})$公佈給所有 $KEA1_i$。
(3)$KEA1_i$ 分別計算 $h_i=(\sum a_k\cdot I_{KEA1_i}^{k})\cdot B=f(I_{KEA1_i})\cdot B=S_{KEA1_i}\cdot B=P_{KEA1_i} \pmod p$，$k=0\sim t-1$，$i=1\sim n$，並驗證 $h_i \stackrel{?}{=} P_{KEA1_i}$。若不成立則將採用廣播的方式告知 $KEA1_i$ 所收到的 S_{KEA1_i} 為偽的。若成立則 $KEA1_i$ 接受此 S_{KEA1_i}。
KEA2 運作金鑰託管過程與上述相同。

2.4 通訊階段

此階段分為三個部份，分別是交談金鑰互換、*LEAF* 產生與驗證、加解密。以下說明其運作過程。
(一)交談金鑰互換機制
使用者 U_a 與 U_b 分別建立其共同的 SK_{ab} 如下：
1.U_a 任選一亂數 $x_a\in[2, q-2]$，計算 $T_a=x_a\cdot B \pmod p$，並將 I_a、P_a 與 T_a 傳送給 U_b。
2.U_b 任選一亂數 $x_b\in[2, q-2]$，計算 $T_b=x_b\cdot B \pmod p$，並將 I_b、P_b 與 T_b 傳送給 U_a。
3.U_a 計算其交談金鑰 SK_{ab} 如下：
$V_b=P_b+h(I_b)\cdot B+(X(P_b)+h(I_b))\cdot P_{SA} \pmod p$
$SK_{ab}=x_a\cdot V_b+s_a\cdot T_b=x_a\cdot s_b\cdot B+s_a\cdot x_b\cdot B \pmod p$
4. U_b 計算其交談金鑰 SK_{ab} 如下：
$V_a=P_a+h(I_a)\cdot B+(X(P_a)+h(I_a))\cdot P_{SA} \pmod p$；
$SK_{ab}=x_b\cdot V_a+s_b\cdot T_a=x_b\cdot s_a\cdot B+s_b\cdot x_a\cdot B \pmod p$
(二)*LEAF* 產生階段
使用者 U_a 與 U_b 進行通訊時，必須事先產生之 *LEAF*，其產生過程如下：
計算 $c=X(SK)\cdot B \pmod p$；計算 $t=(P_{KEA1}+P_{KEA2})\cdot X(SK) \pmod p$；計算 $n=SK+t \pmod p$；則可得出 $LEAF=(c, n)$
(三)加/解密機制
以下列出此機制之運作過程及參數定義。
M 為欲加密之訊息；U_a 為加密者；U_b 為解密者。
[加密]
U_a 利用事先產生的 SK_{ab} 加密 M，得出 $C=E_{SK_{ab}}(M)$，傳送 C 與 *LEAF* 給 U_b。
[解密]
U_b 取得 C 與 *LEAF*。U_b 利用 SK_{ab} 解密即可獲得 M。

2.5 金鑰恢復階段

此階段之運作過程如下：
1.合法授權者向系統中心提出金鑰恢復的請求，並且傳送接收到之$\{C, LEAF\}$給系統中心。
2.系統中心向 KEA1 與 KEA2 提出要求，利用任意 t 個 $KEA1_i$ 與 $KEA2_j$ 回傳之 S_{KEA1i} 與 S_{KEA2j}，用 Lagrange 內插法還原得出 S_{KEA1} 與 S_{KEA2}。
3.系統中心得到 S_{KEA1} 與 S_{KEA2} 後，計算 $w=(-S_{KEA1}-S_{KEA2})=-(S_{KEA1}+S_{KEA2}) \pmod q$
4.再計算
$$SK=(w\cdot c+n) \pmod p \quad (2)$$
即可得出 SK，解密 C 後，得出原訊息 M，傳送

M 給提出請求之合法授權者。

定理 2：系統中心可於 *LEAF* 解密階段重新恢復出交談金鑰 SK。

證明：

$(w \cdot c+n) \pmod p=(-(S_{KEA1}+S_{KEA2}) \cdot X(SK)) \cdot B+(SK+t) \pmod p=-X(SK) \cdot (P_{KEA1}+P_{KEA2})+SK+(P_{KEA1}+P_{KEA2}) \cdot X(SK) \pmod p=SK \pmod p$

Q.E.D

.

2.6 安全性分析

本研究機制之安全性主要是基於解橢圓曲線離散對數問題與單向雜湊函數的困難度。以下將探討機制中各個階段的安全性。

(一)註冊階段

系統中心無法取得使用者的私鑰 s_i，其安全度是基於解橢圓曲線離散對數問題的困難度；攻擊者無法取得使用者的私鑰 s_i 與系統中心的私鑰 s_{SA}，其安全度是基於解橢圓曲線離散對數問題的困難度；使用者無法自行計算有效的公鑰及證明。

(二)金鑰託管階段

因為 KEA1 與 KEA2 會將驗證序列公佈給所有的 $KEA1_i$ 與 $KEA2_j$，所以 KEA1 與 KEA2 無法偽造分享之 S_{KEA1i} 與 S_{KEA2j}。

(三)通訊階段

第三者無法解出交談金鑰，其安全性是基於解橢圓曲線離散對數問題的困難度；攻擊者無法由 *LEAF* 中求出交談金鑰，此安全性為基於解橢圓曲線離散對數問題的困難度；攻擊者截取到 C 與 *LEAF* 後無法解出明文，因為使用者無法取得交談金鑰 SK，其安全度是基於解橢圓曲線離散對數問題的困難度。

(四)金鑰恢復階段

SA 無法自行重新恢復交談金鑰。

2.7 複雜度分析

複雜度分析包含計算時間複雜度與通訊傳輸量兩方面。

(一)計算時間複雜度

為了方便分析，故定義以下參數：

K：一個任意大小的有限場；T_{EM}：進行一次 ECC 乘法運算所需花費的時間(如：$P_i=s_i \cdot B \pmod p$)；T_{EA}：進行一次 ECC 加法運算所需花費的時間(如：$P+Q=R$)；T_{MM}：進行一次模數 K 乘法運算所需花費的時間；T_{MD}：進行一次模數 K 除法運算所需花費的時間；T_{MA}：進行一次模數 K 加法運算所需花費的時間；T_{ME}：進行一次模數 K 指數運算所需花費的時間；T_H：進行一次單向雜湊函數運算所需花費的時間；T_{SYM}：進行一次對稱式加/解密所需花費的時間；T_{INV}：進行一次模數 K 乘法反元素運算所需花費的時間。

相較於模數乘法，模數加法、模數減法、互斥運算時間相當低，故可忽略不記。另外根據文獻[4]所作的假設得知，在橢圓曲線中，計算 $k \cdot B$ 是針對所有的點 $B \in E(F_p)$ 且 $p \approx 2^{160}$，其中 k 是一個隨機的 160 位元整數。此外，對於 $g^x \bmod P$ 的計算，其中 P 是一個 1024 位元的質數，x 是一個隨機的 160 位元整數。於是可以推得出 $T_{EM} \approx 29T_{MM}$、$T_{EA} \approx (5/41)T_{MM} \approx 0.12T_{MM}$、$T_{ME} \approx 240T_{MM}$，且由文獻[3]可知 $T_{MM}=|K|^{\log_2 3} \approx |K|^{1.585}$，$T_{MD} \approx T_{MM}$ 以及 $T_{INV} \approx (0.843*\ln(K)+1.47)*T_{MD}$。

表 2-1 計算時間複雜度及概略估計比較表

項目	Nechvatal [34]系統	本研究機制
系統建置階段	$T_{ME} \approx 240T_{MM}$	$T_{EM} \approx 29T_{MM}$
使用者註冊階段	$(2n^2-1)T_{MM}$	$5T_{EM}+3T_{EA}+T_{MM}+4T_H \approx 146.36T_{MM}+4T_H$
金鑰託管階段	$2T_{ME}+2T_{MM} \approx 482T_{MM}$	$4nT_{EM} \approx 116n\ T_{MM}$
通訊階段	$2nT_{ME} \approx 480nT_{MM}$	$8T_{EM}+6T_{EA}+2T_H+2T_{SYM} \approx 232.72T_{MM}+2T_H+2T_{SYM}$
金鑰恢復階段	$T_{ME}+T_{MM} \approx 241T_{MM}$	$T_{EM}+T_{EA} \approx 29.12\ T_{MM}$

n：子金鑰託管單位之個數。

由表 2-1 可看出本研究機制各階段之時間複雜度皆比 Nechvatal[7]之系統來得少。所以此基於橢圓曲線密碼系統之自我認證公開金鑰密碼系統設計之機制，會比 Nechvatal 的系統更來得有效率，故本研究機制應用到電子商務環境確實兼具效率性及安全性。

(二)通訊傳輸量

為了方便分析，故定義以下參數：

$|M|$：明文大小；$|M'|$：使用交談金鑰加密過的長度；$|I|$：身份相關資訊的大小；$|p|$：經過模 p 運算後的大小，p 為 160 位元的質數；$|q|$：經過模 q 運算後的大小，q 為 160 位元的質數；$|q'|$：經過模 q'運算後的大小，q'為 512 位元的質數；$|X|$：使用者自行選擇的隨機整數大小；$|cert|$：公鑰憑證。

表 2-2 通訊傳輸量比較表

項目	Nechvatal[7]系統	本研究機制
使用者註冊階段	$2n^2\|p\|$	$\|I\|+2\|p\|+\|q\|$
金鑰託管階段		$2n\|q\|$
通訊階段	$2\|p\|+\|M'\|+\|cert\|$	$2\|I\|+4\|p\|+\|M'\|$
金鑰恢復階段	$\|M'\|+\|M\|+2\|p\|+2n\|q'\|$	$\|M'\|+\|M\|+2\|p\|+2t\|q\|$

n：子金鑰託管單位的總數($n>t$)。

t：重新恢復金鑰所需之子金鑰託管單位的個數。

由表 2-2 中可看出，本研究提出之機制僅有在金鑰託管階段的通訊傳輸量大於 Nechvatal 的系統，但以整體來看，本研究機制之通訊傳輸量是少於 Nechvatal 的系統。因此可得知，本研究機制除具有 ECC 的安全性外，亦能增加訊息傳輸的效率性，正可應用於需要快速運算的電子商務環境中。

3. 託管電子現金系統

本研究所設計的託管電子現金系統是基於 Tsaur 與 Ho[11]所提出的電子付款系統，並結合了前述的金鑰恢復與託管機制. 此系統共包含七個主要的階段，以下簡單描述系統組成之要素：

1.系統中心(SA)：協助所有參與者產生其金鑰對。
2.金鑰託管機構(KEA)：託管使用者與商店的秘密金鑰或交談金鑰，以及在必要時協助 SA 恢復金鑰。
3.銀行(Bank)：發行電子現金與清償商家儲存之電子現金。

3.1 系統建置階段

系統中心的初始化，用以建立公開及秘密參數。由 SA 計算以下參數：p、F_p 場中的橢圓曲線 E、B、$h(\ )$、$X(P)$、P_{SA}、s_{SA}、w_{Ui} 與 I_{Ui} 之定義如前 2.1 所述. 當 SA 建立這些參數後, 將$\{E, B, p, q, P_{SA}, h(\ )\}$公開，保留 s_{SA} 當成其密鑰。

3.2 初始階段

此階段包括使用者向銀行開設帳戶，以及所有參與者向系統中心註冊取得金鑰對。另外，資料的傳送是利用雙方產生之交談金鑰將資料加密, 交談金鑰產生方式如前 2.4 所述。以下就各個參與者的參數作說明：

(一)使用者(User)

I_{Ui}：使用者 U_i 的身份；s_{Ui}：使用者 U_i 的秘密金鑰；ACC_{Ui}：使用者 U_i 的帳戶號碼；P_{Ui}：使用者 U_i 的公開金鑰；C-DB：儲存所提領還沒有花費的電子現金的資料庫；PS_x-DB：儲存使用者秘密金鑰及秘密資訊的資料庫。

(二)銀行(Bank)

I_{Bj}：銀行的身份；s_{Bj}：銀行的秘密金鑰；P_{Bj}：銀行的公開金鑰；U-DB：儲存使用者身份以及銀行帳戶號碼的資料庫；W-DB：儲存提款副本(Withdrawal Transcripts)的資料庫；D-DB：儲存清償副本(Deposit Transcripts)的資料庫。

(三)商家(Shop)

I_{Sk}：商家的身份；s_{Sk}：商家的秘密金鑰；P_{Sk}：商家的公開金鑰；ACC_{Sk}：商家的帳戶號碼；P-DB：儲存尚未清償電子現金的付款資料庫；U-BL：記錄廢止的使用者公開金鑰；C-WL：記錄合法的電子現金；B-BL：記錄廢止的銀行公開金鑰；PS-WL：記錄合法的使用者公開金鑰。

(四)金鑰託管機構(Key Escrow Agent)

(P_{KEA1}, s_{KEA1})：KEA1 的公私金鑰；(P_{KEA2}, s_{KEA2})：KEA2 的公私金鑰。

[開設帳戶]

使用者 U_i 使用 I_{Ui} 向銀行 B_j 開設帳戶，得到一組帳戶號碼 ACC_{Ui}。銀行 B_j 儲存(I_{Ui}, ACC_{Ui})在 U-DB 資料庫中。

[使用者註冊]

使用者 U_i 執行以下步驟向 SA 註冊以取得合法公鑰及證明，並計算其密鑰。

1.使用者 U_i 隨選一個亂數 x_{Ui}，且計算 $V_{Ui}=h(x_{Ui}, I_{Ui}) \cdot B \pmod p$，並將自己的身分 I_{Ui} 與 V_{Ui} 傳送給 SA。
2.SA 選擇一個亂數 k_{Ui}, 且計算使用者 U_i 的公鑰為 $P_{Ui}=V_{Ui}+(k_{Ui}-h(I_{Ui})) \cdot B \pmod p$與公鑰證明 $w_{Ui}=k_{Ui}+s_{SA} \cdot (X(P_{Ui})+h(I_{Ui})) \pmod q$，並傳送 P_{Ui} 與 w_{Ui} 給使用者 U_i。
3.使用者 U_i 計算自己的私鑰 $s_{Ui}=w_{Ui}+h(x_{Ui}, I_{Ui}) \pmod q$，並驗證公鑰的有效性是否成立：$s_{Ui} \cdot B=P_{Ui}+h(I_{Ui}) \cdot B+(X(P_{Ui})+h(I_{Ui})) \cdot P_{SA} \pmod p$ 若驗證成功，則使用者 U_i 的私鑰為 s_{Ui}，公鑰為 P_{Ui}。

在註冊階段中，銀行須隨選亂數 x_{Bj}、k_{Bj}；商家須隨選亂數 x_{Sk}、k_{Sk}。而銀行、商家與金鑰託管單位產生公鑰與私鑰的方法皆如同使用者註冊。

3.3 金鑰託管階段

當使用者與商店向 SA 註冊後, 必須將其部份秘密金鑰與交談金鑰託管至 KEA1 與 KEA2 中。

(一)使用者金鑰託管

1.U_i 與 KEA1、KEA2 共同產生交談金鑰 SK_{UK1}、SK_{UK2}，以便在傳輸資料時供加/解密用。
2.U_i 分別將(w_{Ui}, P_{Ui}, I_{Ui})與$(h(x_{Ui}, I_{Ui}), P_{Ui}, I_{Ui})$送給 KEA1 與 KEA2。
3.KEA1 計算 $C2=w_{Ui} \cdot B \pmod p$；KEA2 計算 $C3=h(x_{Ui}, I_{Ui}) \cdot B \pmod p$。並且分別傳送 $C2$ 與 $C3$ 到 SA。
4.SA 收到後，驗證 $C2+C3$ 之值是否等於 $P_{Ui}+h(I_{Ui}) \cdot B+(X(P_{Ui})+h(I_{Ui})) \cdot P_{SA} \pmod p$。

(二)商家金鑰託管

1.商家將其與使用者共同產生之交談金鑰 SK_{SU} 託管至金鑰託管單位。
2.S_k 與 KEA1、KEA2 共同產生交談金鑰 SK_{SK1}、SK_{SK2}，以便在傳輸資料時供加/解密用。
3.S_k 計算 $n=SK_{US}+(P_{KEA1}+P_{KEA2}) \cdot X(SK_{US}) \pmod p$，分別將$(X(SK_{US}), P_{Sk}, I_{Sk})$與$(n, P_{Ui}, I_{Ui})$送給 KEA1 與 KEA2。

3.4 提款階段

使用者執行以下步驟向銀行提領電子現金：

1.使用者 U_i 與銀行 B_j 共同產生 SK_{UB}，在傳輸資料時加/解密用。SK_{UB} 產生方式同前述 2.4.1。
2.U_i 選擇 C 並計算 $C'=h(X(P_{Ui}), C)$, 將 $E_{SK_{UB}}(C', C)$ 傳給 B_j。
3.B_j 收到後，隨機選取一個數 u_{Bj} 並計算 $R_{Bj}'=u_{Bj} \cdot B \pmod p$，將 R_{Bj}'傳給 U_i。
4.U_i 收到後隨機選取兩個數 a 與 b 並計算 $R_{Bj}=R_{Bj}' \cdot a+b \cdot B \pmod p$；$F=h(X(R_B), C, X(P_{Ui}))$；$F'=F/a \pmod q$；將 F'傳給 B_j。
5.B_j 收到 F'後，計算 $S_{Bj}'=s_{Bj} \cdot F'+u_{Bj} \pmod q$，並

將 S_{Bj}'傳給 U_i，然後 B_j 儲存 I_{Ui}, C', s_{Bj} 在 W-DB 資料庫中，並從使用者帳戶中扣款。

6.U_i 收到 S_{Bj}'後計算 $S_{Bj}=S_{Bj}$'．$a+b \pmod q$，$\sigma_{Bj}=(R_{Bj}, S_{Bj})$並驗證 S_{Bj}．$B=P_{Bj}$．$h(X(R_{Bj}), C, X(P_{Ui}))+R_{Bj} \pmod p$

驗證式成立後，則儲存電子現金(C, σ_{Bj}, P_{Ui})在 C-DB 資料庫中。

3.5 付款階段

在付款階段中，使用者將其電子現金，傳送給商家作付款的動作。此階段運作方法如下：

1.使用者 U_i 與商家 S_k 共同產生交談金鑰 SK_{US}，在傳輸資料時加/解密用。SK_{UB} 產生方式同前 2.4。

2.商家 S_k 寄給使用者 U_i 時間戳記 $mess$。

3.U_i 收到後，隨機選取一個數 K_{Ui} 並計算 $R_{Ui}=K_{Ui}$．$B \pmod p$；$S_{Ui}=s_{Ui}$．$h(X(R_{Ui}), C, I_{Ui}, mess, X(P_{Ui}))+K_{Ui} \pmod q$；$\sigma_{Ui}=(R_{Ui}, S_{Ui})$；將 C, P_{Ui}, σ_{Bj}, σ_{Ui} 傳給商家。

4.商家收到 C, P_{Ui}, σ_{Bj}, σ_{Ui} 後檢驗：

假如 $P_{Ui}\in$U-BL，則拒絕電子現金 C；假如 $P_{Bj}\in$B-BL 且 $h(X(P_{Ui}), C)\notin$C-WL，則拒絕電子現金 C；

5.若上述檢驗都沒問題，則驗證下面等式是否成立：

S_{Bj}．$B=P_{Bj}$．$h(X(R_{Bj}), C, X(P_{Ui}))+R_{Bj} \pmod p$；$S_{Ui}$．$B=P_{Ui}$．$h(X(R_{Ui}), C, I_{Ui}, mess, X(P_{Ui}))+R_{Ui} \pmod p$；若等式都驗證成立，則將$(C, P_{Ui}, mess, \sigma_{Ui}, \sigma_{Bj})$儲存在 P-DB 資料庫中，接受使用者的付款。

3.6 清償階段

商家執行以下步驟向銀行要求清償動作：

1.商家 S_k 將電子現金(C, σ_{Bj}, P_{Ui})傳送給銀行 B_j。

2.B_j 收到電子現金(C, σ_{Bj}, P_{Ui})後，驗證以下等式是否成立。S_{Bj}．$B=P_{Bj}$．$h(X(R_{Bj}), C, X(P_{Ui}))+R_{Bj} \pmod p$。若等式成立後，接著檢驗 C 是不是已經清償過了，搜尋 D-DB 資料庫中，假如找到(C, σ_{Ui}')，則代表是重複清償，B_j 回傳 S_k 一個有關電子現金 C 是不合法的回應；如沒找到，則 B_j 傳給 S_k 一個有關電子現金 C 是合法的回應。

3.S_k 再將 I_{Sk}, ACC_{Sk}, $mess$, σ_{Ui} 傳給 B_k。

4.B_k 再驗證下列等式是否成立。S_{Ui}．$B=P_{Ui}$．$h(X(R_{Ui}), C, I_{Ui}, mess, X(P_{Ui}))+R_{Ui} \pmod p$，若等式成立，則檢驗是否有超支，假如超支的話，則執行追蹤階段。假如沒有超支的話，則將錢存入商家的帳戶中，銀行儲存$(C, P_{Ui}, I_{Sk}, \sigma_{Ui})$在 D-DB 資料庫中。

3.7 追蹤階段

(一)使用者追蹤

假如發現超支，則執行以下步驟：

(1)銀行 B_j 將$(C, P_{Ui}, mess, \sigma_{Ui}, \sigma_{Bj})$送給 SA。

(2)SA 向 KEA1 與 KEA2 提出合法授權要求。

(3)KEA1 計算 $C2=w_{Ui}$．$B \pmod p$；KEA2 計算 $C3=h(x_{Ui}, I_{Ui})$．$B \pmod p$。並且分別傳送$(C2, P_{Ui}, I_{Ui})$與$(C3, P_{Ui}, I_{Ui})$到 SA。

(4)SA 收到後，驗證 $C2+C3$ 之值是否等於 $P_{Ui}+h(I_{Ui})$．$B+(X(P_{Ui})+h(I_{Ui}))$．$P_{SA} \pmod p$。

(5)驗證成功後，SA 可得知 I_{Ui}，之後告知 B_j 以便找出是誰超支花費。

(二)違法交易追蹤

假設調查單位懷疑商家進行非法交易時，追蹤步驟如下：

(1)擷取交易的密文 C'，其中 C'是以 SK_{US} 加密$(C, P_{Ui}, \sigma_{Bj}, \sigma_{Ui})$。

(2)調查單位持 C'向 SA 申請解密。

(3)SA 向 KEA1 與 KEA2 提出合法授權的請求。

(4)KEA1 計算 $c=X(SK_{US})$．$B \pmod p$，KEA1 與 KEA2 分別將(c, s_{KEA1}, I_{Sk})與(n, s_{KEA2}, I_{Sk})送到 SA。

(5)SA 計算 $-(s_{KEA1}+s_{KEA2})$．$c+n=SK_{US}$。

(6)為了查出電子現金的使用者，此時調查單位必須再執行使用者追蹤，以便找出電子現金的使用者。

3.8 安全性分析

本研究所提出之託管電子現金系統是以 Tsaur 與 Ho[11]提出之電子付款系統為基礎，加入前面所提出之金鑰恢復與託管機制來設計。其安全性主要是基於解橢圓曲線離散對數問題之困難度與雜湊函數之安全性。

(一)初始階段

系統中心無法取得參與者的秘密金鑰，其安全度是基於解橢圓曲線離散對數問題的困難度；攻擊者無法取得參與者的私鑰 s_i 與系統中心的私鑰 s_{SA}，其安全性是基於解橢圓曲線離散對數問題的困難度；參與者無法自行計算有效的公鑰及證明。

(二)金鑰託管階段

攻擊者無法取得使用者傳送至金鑰託管單位的資料，因為攻擊者無法取得使用者與金鑰託管單位共同產生之交談金鑰 SK_{UK1} 與 SK_{UK2}；攻擊者無法取得商家傳送至金鑰託管單位的資料，因為攻擊者無法取得商家與金鑰託管單位共產產生之交談金鑰 SK_{SK1} 與 SK_{SK2}；由於使用者並非直接託管其金鑰，故可避免託管單位得知使用者金鑰，保護使用者金鑰的安全。

(三)提款階段

電子現金無法被偽造，因為使用者所提領的電子現金是經由銀行簽署過的，所以沒有人可以偽造銀行來發行電子現金。

(四)付款階段

商家可自行驗證電子現金商家可自行驗證，而不用請發行電子現金的銀行來幫忙驗證，此乃由於採用自我認證的方式。此外，在驗證電子現金真偽時，也可以驗證對方的身份及其公鑰有效性。

(五)清償階段

銀行可檢驗電子現金是否已經清償與超支。
(六)追蹤階段
非法交易發生時可追蹤出犯罪者。

3.9 複雜度分析

(一)計算時間複雜度
參數定義如前 2.7 所述。

表 3-1 託管電子現金系統之時間複雜度概略估計

	SA	User	Bank	Shop	KEA1& KEA2
系統建置	$29T_{MM}$				
初始	150.6 T_{MM} ＋ $10T_H$	116.24 T_{MM} ＋ $2T_H$	116.24 T_{MM} ＋ $2T_H$	116.24 T_{MM} ＋ $2T_H$	32.48 T_{MM} ＋ $4T_H$
託管	58.36 T_{MM} ＋ $2T_H$	45.36 T_{MM} ＋ $2T_H$		174.6 T_{MM} ＋ $2T_H$	$87T_{MM}$ ＋ T_H
提款		263.6 T_{MM} ＋ T_{SYM} ＋ $5T_H$	175.36 T_{MM} ＋ $2T_H$		
付款		175.36 T_{MM} ＋ $3T_H$		261.6 T_{MM} ＋ $4T_H$	
清償			261.6 T_{MM} ＋ $4T_H$	145.36 T_{MM} ＋ $2T_H$	
使用者追蹤	58.36 T_{MM} ＋ $2T_H$				$58T_{MM}$ ＋ T_H
違法追蹤	29.12 T_{MM}				$29T_{MM}$

表 3-2 Lee 等學者提出系統之時間複雜度概略估計

	Trustee	User	Bank	Shop
系統建置	$29T_{MM}$			
初始	$264.36T_{MM}$ ＋$6T_H$	$204.12T_{MM}$＋ $2T_H$	$204.12T_{MM}$ ＋ $2T_H$	$204.12T_{MM}$ ＋$2T_H$
提款		$263.36T_{MM}$＋ $(0.843 \times \ln(q)$ ＋$1.47) \times T_{MD}$ ＋T_H	$117.12\ T_{MM}$	
付款		$30T_{MM}$＋T_H		$174.36T_{MM}$ ＋$3T_H$
清償			$58.12T_{MM}$ ＋ T_H	$58.12T_{MM}$ ＋ T_H
使用者追蹤	$174.36T_{MM}$ ＋$3T_H$			

我們比較表 3-1、3-2 後可以發現到，在初始階段及使用者追蹤階段中，本研究機制的計算複雜度是較低的。雖然在其他階段之計算複雜度略大於 Lee 等[5]學者提出之系統，但是本研究機制卻提供使用者更大的安全性。因為我們的機制中另外增添了違法交易追蹤的功能，因此在安全性上是優於 Lee 等[5]提出之系統。並且由於我們的機制在追蹤部分的計算複雜度是較少的，故當犯罪行為產生時，能更迅速地找出犯罪者，還給使用者一個公道。
(二)通訊傳輸量
各參數的定義如下：
1. $|M|$、$|M'|$、$|I|$、$|p|$、$|q|$、$|X|$之定義如前 2.7。
2. $|m|$：時間戳記長度；$|Acc|$：銀行帳號長度。

表 3-3 託管電子現金系統之通訊傳輸量

項目	通訊傳輸總量	
初始階段	$5\|I\|+10\|p\|+5\|q\|$	
金鑰託管階段	使用者金鑰託管	$2\|I\|+6\|p\|+2\|M'\|$
	商家金鑰託管	$2\|I\|+4\|p\|+2\|M'\|$
提款階段	$2\|I\|+5\|p\|+2\|q\|+\|M'\|$	
付款階段	$2\|I\|+7\|p\|+2\|q\|+\|X\|+\|m\|$	
清償階段	$\|I\|+3\|p\|+2\|q\|+\|X\|+\|m\|+\|Acc\|$	
追蹤階段	使用者追蹤	$\|I\|+3\|p\|+2\|q\|+\|X\|+\|m\|+2\|M'\|$
	違法追蹤	$2\|M'\|$

表 3-4 Lee 等學者提出系統之通訊傳輸量

項目	通訊傳輸總量	
初始階段	$3\|I\| + 12\|I\| + 6\|I\|$	
提款階段	$2\|p\| + 2\|q\|+\|M'\|$	
付款階段	$8\|p\|+3\|q\|+\|X\|+\|m\|$	
清償階段	$\|I\|+10\|p\|+4\|q\|+\|X\|+\|m\|+\|Acc\|$	
追蹤階段	使用者追蹤	$\|I\|+12\|p\|+4\|q\|+\|X\|$

理論上來說$|I|$會小於$|p|$與$|q|$，因此由表 3-3、3-4 可得知，我們的系統在初始、提款、付款與清償階段之傳輸量皆小於 Lee 等[5] 學者提出之系統，只有在追蹤階段之傳輸量略大於 Lee 等學者提出之系統。此外本研究機制中多了金鑰託管單位的存在，故會增加金鑰託管階段的傳輸量，但是因為這個階段的存在，我們才能提供使用者更安全的環境，並且讓犯罪者能更快的經由追蹤階段被找出。因此，本研究機制除了具有 ECC 的安全性與效率外，提供之追蹤階段更能幫助執法單位快速找出犯罪者，讓使用者能更安心的在電子商務環境中進行交易。

4. 結論

本研究提出之兩個機制皆應用自我認證的安全機制，使得使用者不必依賴系統中心就能達到身份確認，因為可以在同一個邏輯步驟內完成身份驗證與公鑰的正確性檢查，故能提高效率性。而在安全性方面，本論文提出之兩個機制因為是植基於橢圓曲線密碼系統來發展，故可以更少的位元數來達到相同的安全等級，因此適合應用於需要快速運算

的電子商務環境上。

5. 參考文獻

[1] 陳宗保，「行動電子商務環境下安全協定之研究」，大葉大學資訊管理研究所碩士論文，民國 90 年。

[2] D. E. Denning and M. Smid, “Key escrowing today,” IEEE Communications, Vol. 32, pp. 58-68, 1994.

[3] D. F. Knuth, “Seminumerical Algorithms,” The Art of Computer Programming, Addison-Wesley, Vol. 2, 1981.

[4] N. Koblitz, “Elliptic curve cryptosystems,” Mathematics of Computation, Vol. 48, No. 17, pp. 203-209, 1987.

[5] M. Lee, G. Ahn, J. Kim, J. Park, B. Lee, K. Kim, and H. Lee, “Design and implementation of an efficient fair off-line E-Cash system based on elliptic curve discrete logarithm problem,” Journal of Communications and Networks, Vol. 4, 2002.

[6] W. Mao, “Publicly verifiable partial key escrow,” International Conference on Information and Communications Security, Springer-Verlag, LNCS , pp. 409-413, 1997.

[7] J. Nechvatal, “A public-key-based key escrow system,” Journal of Systems and Software, Vol. 35, pp. 73-83, 1996.

[8] J. M. Nieto, K. Viswanathan, C. Boyd, and E. Dawson, “Key recovery system for the commercial environment,” International Journal of Information Security, Vol. 1, pp. 161-174, 2002.

[9] T. P. Pedersen, “Distributed provers with applications to undeniable signature,” Advances in Cryptology–EUROCRYPT’91, LNCS, Vol. 547, Springer-Verlag, pp. 221-238, 1991.

[10] A. Shamir, “Partial key escrow：A New Approach to Software Key Escrow,” NIST Key Escrow Standards meeting, 1995.

[11] W. J. Tsaur and C. H. Ho, “A Secure Electronic Payment System Based on Efficient Public Key Infrastructure,” Proceedings of the 2002 International Workshop for Asian Public Key Infrastructures (IWAP 2002), Taipei, Taiwan, 2002.

[12] K. Viswanathan, C. Boyd and E. Dawson, “Strong binding for software key escrow,” International Workshops on Parallel Processing, IEEE Press., 1999.

[13] H Wang, and Y. Zhang “Untraceable off-line electronic cash flow in e-commerce,” Proceedings of Computer Science Conference, pp. 191-198, 2001.