修 士 論 文

インターネット上での通貨に関する研究

指導教官　篠田 陽一 助教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

三輪 信介

1997 年 2 月 14 日

目 次

第 1 章

はじめに

近年爆発的にその接続利用者数を増やしているインターネットにおいて、情報やソフトウェアを買うといった小額の取引のみならず、パソコンや携帯電話など一般的で高額な商品を取引する機会も増えてきている。

インターネット上でのオンラインショッピングは、一般的に 24 時間、どこの国からでもインターネットに接続可能であれば利用できる。そのため、24 時間決済や国際決済といった現在の通貨や決済方式だけでは、対応が難しい状況が生じてきた。そこで、これらの状況において有効な決済システムが望まれている。

このような状況は、インターネットが普及する以前から、経済の国際化によって生じてきている。そこで現実の社会においては、24 時間決済や国際決済のための能力を高め、決済コストを減らすことができる決済システムの研究が行われており、最近ではいくつかのシステムが実験的に特定の都市や特定の小売店、特定の商品への決済で利用できるようになってきている。

インターネット上での決済システムも同様に研究が進んでおり、最近ではいくつかのシステムが実現されている。しかし、多くのシステムは決済することに重きがおかれており、実際には決済情報のやり取りに過ぎないものが多い。これらのシステムでは、表面上は 24 時間決済や国際決済の要求を満たすが、最終的決済が 24 時間かつ国際的に行えるわけではない。また、最終的決済の際に決済コストがかかり、利用者や決済機関の負担が増加するという問題がある。

このような要求を真に満たすためには、インターネット上で最終的決済が行えなければならない。そのためには、インターネット上で 24 時間かつ国際的に流通することがで

きる通貨が必要となる。つまり、インターネット上での通貨が必要となると考えられる。そこで、本研究ではインターネット上での決済システムの現状を踏まえた上で、インターネット上での通貨の要件を考え、新たな決済システムを提案／設計／実装した。その上で、インターネット上での通貨を作り上げる上で何が問題となるのかについて議論し、その展望を考察した。

第 2 章

インターネット上での通貨の要件と現状

インターネット上での通貨を考える上で、いくつかの要素があると考えられるので、ここでは「通貨としての要素」「安全性に関する要素」「普及に関する要素」に分けて、通貨の要件を考える。その上で現状を分析し、問題点とそれに対する提案を行う。

2.1 通貨としての要素

2.1.1 兌換性・不換性

通貨は、その価値によって大きく二つの種類に分けられる。通貨自身が決済の際に、充分な価値を持つ場合と持たない場合である。充分な価値を持たない場合、通貨は最終的決済を行うために価値のある他の何かに交換（兌換）する必要がある。金本位制における通貨がこれに値し、兌換通貨と呼ぶ。これに対し、現在のように通貨自身が最終的決済を行うに充分な価値を持つ通貨を不換通貨と呼ぶ。

兌換通貨においては、その通貨で必ず最終的決済が行えること、つまり必ず兌換できることが通貨の信用にとって重要な要件となる。これを通貨の兌換性と呼ぶ。安定したレートで兌換できることが兌換通貨には求められ、兌換のレートが安定していれば安定しているほど兌換性が高いと考える。兌換通貨においては兌換性の高さが、通貨の信用の高さにつながるといえる。兌換性を高めるためには、兌換を社会的に信用の高い機関が保証する必要がある。

不換通貨においては、その通貨で必ず最終的決済が行えること、つまりどのような場面

においても他の何かに交換すること無しに決済できることが通貨の信用にとって重要な要件となる。これを通貨の不換性と呼ぶ。どんなものに対する支払いでも行えることが不換通貨には求められ、その支払いの対象が広ければ広いほど不換性が高いと考える。不換通貨においては不換性の高さが通貨の信用の高さにつながるといえる。不換性を高めるためには、通貨の発行を社会的に信用の高い機関が行い、同時にその通貨が市場で流通し、実際の市場での決済を行える必要がある。

兌換通貨は、兌換を保証する機関が存在すれば比較的容易に実現できる。しかし、不換通貨は発行する機関が存在しても市場で流通し、決済できる必要があるので、容易ではない。そのためか、現在のインターネット上で実現されている電子通貨システム [17][19][21][10] は、銀行などの強信用機関が一般の通貨への兌換を保証した兌換通貨である。

2.1.2 匿名性

通貨で決済ができることを保証するためには、同じ通貨は誰がどんな目的に使用しようとも、同じ価値を持たねばならない。このためには、通貨は利用者を特定せず、無差別に利用できる必要がある。つまり、ある通貨を支払ったとしてもそれによって誰が何のために支払ったのかを知られることはない。また、その通貨を誰が何のために利用してきたのかを知ることはない。これを通貨の匿名性と呼ぶ。

この通貨の匿名性は、決済における利用者のプライバシーの保護をも可能とする。これは、その通貨を利用者が利用するかどうかを決める際に重要な要素となる。なぜなら、通貨はどんな目的に使用しても同じ価値として利用できる必要があるため、利用者は人に知られたくない取引にも通貨を利用したいと考える。この目的を達成するためには、通貨の匿名性が必要となるからである。

以上のように、決済ができることを保証するためと決済における利用者のプライバシー保護を可能とするために、通貨には匿名性が必要となる。現在インターネット上で実現されている電子通貨システム [3][17] は、この通貨の匿名性に重きをおいたものが多い。しかし、匿名性は同時にその通貨の犯罪への利用やマネー・ロンダリング（資金浄化）への利用などを引き起こす可能性があるため、必要なときに匿名性を無効化できる電子通貨システム [2][1][8][11] が最近では提案されている。

2.2　安全性に関わる要素

2.2.1　暗号化と電子署名

インターネットの通信のためのプロトコルスイートである TCP/IP は、ネットワークの接続性とデータの到達性を提供するが、保証するわけではない。そのため、ネットワークの接続が切れることやデータが到達しないことが考えられる。また、到達するデータの中身を保証するわけではなく、インターネットに対する様々な攻撃法が存在するため、通信内容の盗難、盗聴、改ざんやなりすましが行われ得ることが指摘されている。つまり、インターネットは安全性を提供するわけではない。

そのため、安全性を求められるインターネット上のシステムは、通信路以外で安全性を提供することを考えなければならない。そこで、通信の暗号化や電子署名が必要となる。つまり、内容を人に知られてはならない場合には、暗号化を行い、内容を人に改ざんされてはならない場合には電子署名を施すのである。もちろん、これだけで安全性が確保されるわけではないが、通信路において提供されていない安全性を提供するためには、少なくとも通信の暗号化や電子署名が必要となる。

2.3　普及に関わる要素

2.3.1　ユーザーインターフェイス

一般の通貨の決済は簡単である。支払人は支払いに使用したい通貨を受取人に手渡せばよい。ある通貨システムによる決済が煩雑であった場合、利用者はより煩雑ではない通貨システムを使用したいと考えるだろう。つまり、通貨システムによる決済が利用者にとって容易であればあるほど、そのシステムは普及しやすいと考えることができる。

通貨システムによる決済を煩雑にする要素は二つ考えられる。一つは、通貨システムの決済手続きであり、もう一つは通貨システムのユーザーインターフェイスである。そのため、電子通貨システムの普及のためには、通貨システム自身が煩雑な手続きを必要としないことと、利用しやすいユーザーインターフェイスが提供されていることが求められると考えられる。

2.3.2 スケーラビリティ

通貨の必要な流通量は、時間当たりの取引量から決定することができる。時間当たりの取引量は、利用者が増加し、決済が活発に行われれば行われるほど増加する。そのため、利用者が増加した場合、通貨の流通量と取引量が増加する。

インターネットは近年その利用者数が急増しており、今後しばらくはこの勢いが衰えることはないと思われる。そのため、インターネット上での通貨は、この利用者を支えられるだけの流通量と取引量を確保できなければならない。しかし、インターネット上での通貨システムは、インターネットを介した情報のやり取りによって成り立っているため、流通量や取引量の増大は、通信負荷につながると考えられる。この通信負荷に耐えられるだけのスケーラビリティが、インターネット上での通貨システムには求められると考えられる。

2.4 現在のインターネット上での決済システム

現在のインターネット上での決済システムについて、メカニズムと不正検出の方式の2点から分類し、問題を把握した上で、本研究における提案方式を位置づけることとする。

2.4.1 決済メカニズムによる分類

近年提案もしくは公表もしくは実装されているインターネット上での決済メカニズムは以下の3つのグループに分けることができる。

安全なクレジットカード番号の提示を用いる方法

安全なクレジットカード番号の提示を用いる方法 [18][16] では、やり取りするのは暗号化されたクレジットカード番号と取引情報だけなので、特別な登録を必要としないため、安全な方法であれば、普通にクレジットカードで買い物をする感覚で使えるので、普及の面で有利である。

ただし、決済はクレジットカードで行うので決済能力がクレジットカードのそれに束縛されてしまう。そのため、実際には24時間決済の要求に応えることができず、国際決済能力もクレジットカードのそれに準じてしまうという欠点がある。また、すべての取引に

取引サーバが介在せねばならず、取引量にしたがってサーバの通信負荷が増大するという欠点がある。

電子通貨方式

電子通貨システム [17][21][19] は、普通、発行局から証明された通貨を支払いに使う。通貨の証明を受ける際には、ブラインド署名などを利用して、誰がその通貨の証明を得たかを発行局にはわからないようにする。これによって、電子通貨システムは匿名性を提供することができる。

しかし、二重使用を避けるために大量の取引データを管理しなければならない。また、不正検出を厳格にするために決済のトランザクションにオンラインチェック（第 2.4. 2節）を含んでいるため、すべての取引に発行局が介在せねばならず、取引量にしたがって発行局の通信負荷が増大するという欠点がある。

電子小切手方式

電子小切手システム [22] は、基本的には決済情報だけをやり取りする。例えば、支払人 A から受取人 B に金額 C を支払う場合、A は決済機構に「B へ C 支払う」という情報を渡し、決済機構は A の口座から B の口座に C 移動し、B に「A から C 支払いがあった」と通知する。つまり、電子小切手システムは、電子通貨システムのように大量の取引データを管理する代わりに、各利用者の口座情報のみを管理すればよいことになる。これによって、電子小切手システムは会計監査能力を提供することができる。

しかし、決済機構がすべての取引を管理するので匿名性は提供されない。また、他の方式同様、すべての取引に決済機構が介在せねばならないため、取引量にしたがって決済機構の通信負荷が増大するという欠点がある。

電子通貨にせよ電子小切手にせよ、現在の方式では現実の通貨に転換する必要があり（兌換通貨）、そのやり方やタイミングについて充分に考慮する必要がある。

2.4.2 不正検査の方式による分類

近年提案もしくは公表もしくは実装されている電子通貨方式の不正検査の方式は以下の 3 つのグループに分けることができる。

オンラインチェック方式

オンラインチェック方式とは、受取人が発行局に通貨の正当性を問い合わせるか、発行局が正当性を確かめた通貨を受取人に直接渡すかする方式のことである。つまり、オンラインチェック方式では、通貨の検査は必ず発行局によって行われる。比較的容易に強力な不正検出力を得られるので、現在インターネット上で実現されている電子通貨方式のほとんどがこの方式である。

オフラインチェック方式

オフラインチェック方式とは、受取人が自分で通貨の正当性を確かめる／判断する方式のことである。インターネットのような安全ではない通信路を用いる場合には、情報が改ざんされる恐れがあるためオフラインチェックは必要不可欠であるが、同時に情報の正当性が保証できないのでオフラインチェックだけでは、通貨の正当性を保証することは難しい。そのため、現在インターネット上で実現されている電子通貨方式のほとんどはこの方式を単体では採用していない。

事後検出方式

事後検出方式とは、利用者が不正使用していなければ何もせず、不正使用していた場合には、不正使用の証拠をその利用者が否定できないような形で提出できるようにすることで、不正が行われた後に責任を遡及する方式である。この方法は不正使用を防ぐことはできないが、事後に必ずその事実を証拠付きで示すことができるため、不正使用に対する大きな抑止力を持つといえる。また、犯罪への利用が行われた場合やマネー・ロンダリングへの利用が行われた場合などに、証拠付きで事実を示すことができるため、犯罪への利用に対する抑止力も期待されている。この方法と前述の 2 つを組み合わせることで、譲渡機能を電子通貨に付加することができる可能性があるので、最近の電子通貨方式では、この方式を採用するものが多くなってきている。

2.4.3 現在のインターネット上での決済の問題点

現在のインターネット上での決済システムは、どの方式でも何らかのサーバが全ての取引に介在する。これは、今後インターネット上での商取引が活発化し、決済取引量が増

大した場合、サーバに対する通信負荷が大きくなりすぎるという問題がある。特に、電子通貨システムにおいて通貨をトランザクション内においてオンラインチェックする場合、サーバには大きな負担がかかることが考えられる。すなわち、スケーラビリティの面に問題があるといえる。

また、現在のインターネット上での電子通貨システムは、兌換通貨としての機能しかなく、インターネット上で最終的決済ができるわけではない。そのため、24 時間決済や国際決済の要求に必ずしも応えられないといった問題や、最終的決済との間のレートがインターネット上での価値とはかみ合わない可能性があるといった問題、場合によっては一般的な決済手段を用いるよりも決済コストがかかる結果となり、存在意義が薄れてしまうという問題がある。

本研究においては、電子通貨システムに譲渡機能を付加することで、インターネット上で転々譲渡されることにより、真の意味での「インターネット上での決済」が可能となるのではないかと考える。そこで、上記二つの問題の解決への足がかりとして、サーバが介在せずに譲渡することができる機能を付加したインターネット上での電子通貨システムを考えることとする。そのために、不正の事後検出を行うこととし、オフラインチェックと組み合わせることで不正に対する充分な抑止力を持つようにする。

第 3 章

裏書譲渡可能な銀行券方式

現在のインターネット上の電子決済システムの欠点を克服するために、インターネット上で転々譲渡することができる電子通貨を提案する。これによって、最終的にはインターネット上での不換通貨を目指す。

3.1 概略

本方式では、電子通貨方式の欠点である大量の取引情報の管理と電子通貨／電子小切手両方式の欠点である発行局／決済機構への通信負荷の増大を避けるために、手形の裏書機能を電子通貨方式に付加し、不正の事後検出を行うことで、発行局が介在しない転々譲渡を可能とする。

具体的には、銀行券を手形同様に裏書譲渡するようにし、銀行券の使用者を特定できるようにすることで、発行局が大量の取引情報の管理する必要をなくす。また、裏書譲渡の際には発行局と通信する必要をなくすことで、発行局への通信負荷を軽減することができる。不正は、裏書譲渡の署名の連鎖の乱れをもとに事後検出する。検出された不正による損失を補填するために、不正者に対し署名をもとに責任を遡及する。

しかし、単純な裏書では、譲渡の際に使用者がその銀行券の他の全ての使用者を知ることが可能となる。すなわち、匿名性を提供することができない。そこで、本方式では通常の裏書譲渡とは異なり、銀行券の発行局以外は使用者を知ることはできないようにする。これにより、発行局に対する匿名性は提供することはできないが、一般の利用者に対する

匿名性を提供することができる[1]。発行局では銀行券の使用者を特定することができるので、銀行券の犯罪への利用やマネー・ロンダリングへの利用に対する抑止力となると同時に発生後の対応を可能とする。

3.2 モデル

以下に、本方式が想定する世界観とモデルを示す。

3.2.1 想定する世界観

モデルを構成する要素は、実際に通貨の支払いや受取りをする「利用者」、利用者の要求にしたがって通貨を発行する「発行局」、それらをつなぐ「通信路」である。

1. 第三者は悪意を持つ
2. 「利用者」は悪意を持つ可能性がある、
3. 「発行局」は悪意を持たない
4. 「公証局」は安全である

と世界観を想定する。また、インターネットのプロトコルスイートである TCP/IP は安全とはいえないので、「通信路」においては盗難、盗聴、改ざん、なりすましがありうるとする。

以上のような世界観に従うと、すべての通信は盗聴、改ざん、なりすましから守るために暗号化や電子署名を施す必要がある。そこで、各トランザクションにおいて、非対称鍵暗号系 (公開鍵暗号系) を用い、公開鍵による暗号化と、秘密鍵による電子署名を行う。また、replay 攻撃を防ぐために、各通信において直前の通信の内容が反映されるようにする。暗号系は充分な強度を持つとし、秘密鍵の安全性は保持されているとする。

公開鍵の安全で確実な配送のために「公証局 (CA) 」が必要となるが、この公証局は人と鍵を確実に結び付けるだけの充分な強さを持つものとする。銀行券の決済に関わる者は全て公証局に登録することが必要で、銀行券の発行要求と現金化を行うものは発行局に登

[1] 発行局に対する匿名性の提供に関する議論については、第 5.1 節を参照のこと

録する必要がある。この作業を全てインターネット上で行うのは難しいと思われるので、別な通信路を用るなどの必要があると考えている。

また、本方式では、遡及によって支払人に支払いを要求することができなければならないので、法によって手形のように遡及権が確立されている必要がある。

本方式は、決済のみを対象としており、商品を買った／買わないといった問題や、商品が届いた／届かないという問題は別のシステムによって管理されていると仮定する。

3.2.2 裏書譲渡可能な銀行券

第 3.2.1 節の想定に従い、電子的な銀行券を裏書譲渡するようなモデルを考える。銀行券の信用は、発行局が証明することによって与えられる。銀行券は裏書譲渡によって転々譲渡されていき、裏書署名の連鎖が銀行券に記録される。この連鎖を確かめることで、銀行券の正当性を事後的に検証する。

銀行券は、利用者の要求にしたがって「発行局」である銀行などの強信用機関によって発行され、手形同様に裏書譲渡によって利用者同士の間での決済に用いられる。銀行券を受け取った利用者は、任意の時点で正当性の検証や現金化を要求することができる。発行局は、裏書署名の連鎖から銀行券の正当性を事後的に検証する。検証の結果不正が発覚した場合、署名から不正者を特定し、不正者に対して責任を遡及する。これによって、不正を抑止すると同時に不正による損失を充当する。

裏書譲渡の際には、利用者は署名を行うが、この署名が他の利用者に見えないようにするためと内容が改ざんされないようにするために、署名後には発行局の公開鍵で暗号化する。これにより、発行局以外はその内容を知り得ず、正しい内容を再現できないので改ざんを防ぐことができる。発行局は、暗号化を解きながら、署名の連鎖を追うことで、銀行券の事後検証が可能となる。このようなことを可能とするために、銀行券実体は、額面情報と、裏書による署名と暗号化が連なった裏書情報から構成される（図 3.1）。

この方法では、裏書情報は利用者が構成することとなる。そこで、決済を行なうどちらか一方が不正を働くことのないように銀行券の裏書情報は一方だけでは構成できないようにする。つまり、裏書情報は必ず、支払う利用者が受取る利用者との間で取引をしたことを証明する情報と、受取る利用者が支払う利用者との間で取引をしたことを証明する情報の両方から成るようにする。そして、必ず相手が自分と取引したことを証明する情報を自分が書き込み、相手が改ざんできないように発行局の公開鍵で暗号化する。このように

すると、どちらか一方が不正な裏書情報を書き込んだ（もしくは書かなかった）としてももう一方が書き込んだ裏書情報から不正者を特定することができる。

以下にこのモデルに従った本方式における決済取引（図 3.1 ）を示す。

本方式における取引

本方式における取引は、「振出」、「裏書譲渡」、「検証」、「引換」の 4 つのトランザクションから成り立っている。銀行券を受けとったものは任意の時に「検証」を行なうことができ、「引換」の際には必ず「検証」が行なわれる。また、銀行券を任意の額面の銀行券に「分割」することができ、その際にも「検証」が行なわれる。「検証」の際に二重使用などの不正が発見された場合、どの時点でそれがなされたのかが特定され、不正者に責任が遡及される。

銀行券の「振出」は、振出人の要求にしたがって、「発行局」が振り出すことである。「発行局」は銀行などの強信用機関であるとする。決済は、この発行局より振り出された銀行券を支払人が受取人に対して「裏書譲渡」することにより行なわれ、受取人が「引換」することによって現金化される。

「振出」、「裏書譲渡」の際には銀行券に署名を行なう。署名が行なわれないかぎり、「振出」や「裏書譲渡」は行なえない。複製して二重使用した場合、署名を追うことにより、「検証」の際に責任の遡及ができる。

「裏書譲渡」は、公開鍵の交換がなされれば、発行局を介さずに行なうことができる。受取人は任意の時に「検証」ができるので、受けとった銀行券を必ずしも「検証」する必要はない。

銀行券を利用するものは「発行局」と「自分に銀行券を渡した支払人」と「自分が銀行券を渡した受取人」の三者以外について知ることはない。発行局は「検証」の際にその銀行券に関わった全ての者を知ることができる。

3.3　各トランザクションの詳細

各トランザクションの詳細についてみてみよう。前提として、公開鍵は公証局 (CA) を通して安全に交換可能で、ある公開鍵と対を成す秘密鍵を持つのはその公開鍵を登録した人のみであるとする。以下のトランザクションでは公証局を介した公開鍵の交換方法は省

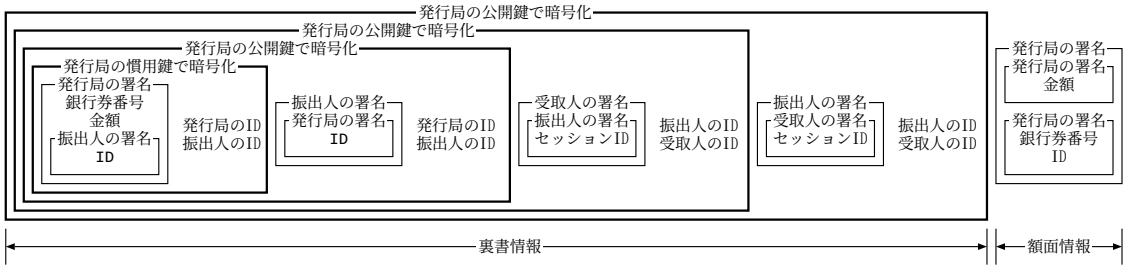


図 3.1: 一度裏書譲渡された後の銀行券の構造

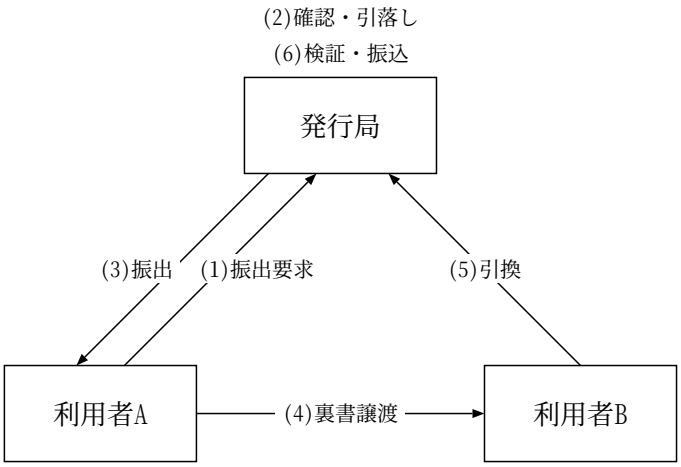


図 3. 2: 取引の概念図

略してある。また、異常時のトランザクションについては詳細を省略してある。

使用されている記号の意味を説明する。$(M)^{K_A}$はメッセージ M の A の慣用鍵による暗号化を示し、$(M)^{PK_A}$はメッセージ M の A の公開鍵による暗号化を示し、$(M)^{SK_A}$はメッセージ M の A の秘密鍵による署名を示す。

以下に、「振出」と「裏書譲渡」と「検証・分割」、「引換」の際のやり取りを示す。

3.3.1 振出

銀行券の振出を要求する人を「振出人」とし、実際の振出を行うものを「発行局」とする。

トランザクションにおける留意点

銀行券番号は連続番号で、すべての銀行券について一意に決定される。これに対し、ID は銀行券毎に違うことが期待される乱数であるが、正確には「ある人が一生のうちに関わる銀行券」に対して、唯一の値が与えられる程度には充分な強度のある乱数であるとする。つまり、ある人が自分の関わったある銀行券を特定するためには ID で充分であるが、発行局や裁判所がある銀行券を特定するためには銀行券番号が必要となるということである。

トランザクションの概略

「振出」のトランザクションの概略は以下のとおり。

署名した ID（振り出す毎に毎回違う情報）を交換し、銀行券の裏書情報として、互いに相手の署名した ID に自分の署名を施したもの（その相手とその ID で取引したことを証明する情報）を書き込んでおく。これにより、相手が自分と取引したことを否定できないようにする。振出人は最後に確認情報として、受け取った銀行券の額面情報に自分の署名を施して送る。

銀行券が届かなかった場合、その旨発行局に振出人は通知する。通知を受けた発行局はその銀行券が使用不能であることを記録し、振出人の口座からの引き落としを行わない。銀行券が届いていたにもかかわらず、振出人が届かなかったふりをした場合は「検証」の際に使用不能の銀行券が使用されたことが発覚するので、通常の不正使用と同様に責任を遡及する。

トランザクションの詳細

1. 公開鍵証明の交換を行なう。

2. 振出人は要求する金額に署名をして（振出人にしか作れない情報)、暗号化して（発行局しかその内容を知ることはできない）送る。

$$\text{振出人} \rightarrow \text{発行局} : ((\text{金額})^{SK_{\text{振出人}}})^{PK_{\text{発行局}}}$$

3. 発行局は暗号化を解き、署名を検証し、残高が金額より大きいことを確認する。

4. 銀行券番号（連番）と ID（乱数）を生成し、ID に署名をして（発行局にしか作れず、かつ毎回違う情報)、暗号化して（振出人しかその内容を知ることはできない）送る。銀行券番号と金額、ID の結び付きを記憶しておく。

$$\text{振出人} \leftarrow \text{発行局} : ((ID)^{SK_{\text{発行局}}})^{PK_{\text{振出人}}}$$

5. 振出人は暗号化を解き、署名を検証する。

6. ID に署名をして（振出人にしか作れず、毎回違う情報)、暗号化して（発行局しかその内容を知ることはできない）送る。

$$\text{振出人} \rightarrow \text{発行局} : ((ID)^{SK_{\text{振出人}}})^{PK_{\text{発行局}}}$$

7. 発行局は暗号化を解き、署名を検証し、ID を確認する。

   ここまでの段階で、振出人と発行局にはそれぞれ相手の署名がなされた ID を入手している。

8. 銀行券番号と金額と振出人の署名がなされた ID に署名をし（振出人からの情報を基に作る発行局にしか作れない情報)、発行局の ID と振出人の ID （後で検証できるように）とともに慣用鍵で暗号化して（発行局のみが作ることができる、その内容を検証できる「裏書情報」)、金額に署名したものと ID と銀行券番号に署名したものに署名し（発行局にしか作れない「額面情報」。金額、ID と銀行券番号は二重署名、後で裏書譲渡の際に金額に発行局の署名を施したものは金額提示用に使い、

ID と銀行券番号は領収書に使うため)、暗号化して(使用することができるのは振出人のみ)送る。

$$\text{振出人} \leftarrow \text{発行局} : (((\text{銀行券番号}, \text{金額}, (ID)^{SK_{\text{振出人}}})^{SK_{\text{発行局}}}, \text{発行局の } ID, \text{振出人の } ID)^{K_{\text{発行局}}}, ((\text{金額})^{SK_{\text{発行局}}}, (ID, \text{銀行券番号})^{SK_{\text{発行局}}})^{SK_{\text{発行局}}})^{PK_{\text{振出人}}}$$

9. 振出人は暗号を解き、署名を検証し、ID と金額を確認する。

10. 受けとった情報の裏書情報部分(発行局の慣用鍵で暗号化されている部分)に加えて、発行局の署名がなされた ID に署名をし(発行局からの情報を基に作る振出人にしか作れない情報)、発行局の ID と振出人の ID(後で検証できるように)とともに発行局の公開鍵で暗号化して(発行局のみがその内容を検証できる)、新しい裏書情報とする。これに、額面情報を加え、銀行券実体として保存する。

$$\text{振出人} : (((\text{銀行券番号}, \text{金額}, (ID)^{SK_{\text{振出人}}})^{SK_{\text{発行局}}}, \text{発行局の } ID, \text{振出人の } ID)^{K_{\text{発行局}}}, ((ID)^{SK_{\text{発行局}}})^{SK_{\text{振出人}}}, \text{発行局の } ID, \text{振出人の } ID)^{PK_{\text{発行局}}}, ((\text{金額})^{SK_{\text{発行局}}}, (ID, \text{銀行券番号})^{SK_{\text{発行局}}})^{SK_{\text{発行局}}})$$

11. 振出人が確実に銀行券を受けとったことを知らせるために、額面情報に署名し(額面情報を基に作る振出人にしか作れない情報)、暗号化して(銀行のみがその内容を知ることができる)送る。

$$\text{振出人} \rightarrow \text{発行局} : ((((\text{金額})^{SK_{\text{発行局}}}, (ID, \text{銀行券番号})^{SK_{\text{発行局}}})^{SK_{\text{発行局}}})^{SK_{\text{振出人}}})^{PK_{\text{発行局}}}$$

12. 発行局は暗号を解き、署名を検証し、額面情報を確認する。振出人の口座から額面分を引き落とす。

3.3.2 裏書譲渡

銀行券を裏書譲渡して支払いを行う人を「支払人」とし、裏書譲渡を受けるものを「受取人」とする。また、その銀行券を振り出したものを「発行局」とする。

トランザクションにおける留意点

「裏書譲渡」のトランザクションは原理的には「振出」のトランザクションと同様である。しかし、単純に同じ方法を取ると二つの大きな問題が生ずる。一つは、「ID の再利用」であり、もう一つは「代金到達の確認」である。

ID の再利用　まず、「ID の再利用」の問題について述べる。「裏書譲渡」においても「振出」と同様に署名した ID を普通に交換した場合を考える。ある特定の銀行券を一度でも「裏書譲渡」したことのある人は、必ず誰かの署名が施された ID と額面情報を持っている。つまり、この署名が施された ID は、その銀行券をその署名をした人と一度でも取引をした人は全員持っていることになる。そのため、この署名が施された ID を不正に再利用できる可能性がある。

例えば、支払人 A が受取人 B に「裏書譲渡」し、受取人 B の署名が施された ID を手に入れたとする。支払人 A はこの受取人 B の署名が施されたＩDと銀行券本体を何人かの共犯者にばらまき、共犯者は受取人 B から「裏書譲渡」を受けたかのようにして、受取人 B の署名が施された ID に自分の署名を施し、裏書情報に追加する。これによって、受取人 B が自分の複製使用を隠すために相手の署名が施された ID に自分の署名を施したものを裏書情報に追加しなかったのか、結託による不正使用が行われたのか区別がつかなくなってしまう。つまり、署名の連鎖を乱し、検証によって不正者を特定することができなくなる。このような不正使用が可能であった場合、受取人から署名付きの ID だけを受け取って、銀行券本体を渡さないという悪質な不正が可能となるなど、大きな問題となる。

そこで、「裏書譲渡」においては ID ではなく、取引（セッション）ごとに違うセッション ID を生成し、署名したセッション ID を交換することにした。セッション ID は、支払人／受取人どちらか一方だけでは生成できず、かつもとの銀行券の ID を含むことで、他の銀行券に関するセッション ID と同じものにならないようにする。具体的には、支払人／受取人ともにセッション毎に乱数を生成し、署名を施す。この署名付きの乱数ともとの銀行券の ID を合わせて、セッション ID とする。このセッション ID を導入することで、「ID の再利用」の問題を解決する。

$$\text{セッション } ID\text{: } ID, (\text{支払人の乱数})^{SK_{\text{支払人}}}, (\text{受取人の乱数})^{SK_{\text{受取人}}}$$

代金到達の確認　次に、「代金到達の確認」の問題を考える。「振出」の場合、発行局が信用できるため発行局がトランザクションにおいて銀行券本体を送らないという場合を考える必要がなく、また、銀行券本体が届けばそれ以上の取引は存在しないため、銀行券本体の到達の確認にさほど留意する必要はない。

しかし、「裏書譲渡」の場合は、支払人／受取人ともに信用できないため、銀行券本体を支払人が渡さずにトランザクションを終了するといった不正や、受取人が銀行券本体を受け取ったにもかかわらず確認情報を送らずにトランザクションを終了するといった不正

がありうる。また、代金の到達の前か後に商品が届いて実際の取引が終了するため、商品が届いたのに代金を支払わない、代金を支払ったのに商品が届かないなどの問題が起こった場合に、本当に代金が支払われたのかどうかの確認をするためにも、代金到達、すなわち銀行券の到達の確認は重要である。

「振出」の方法に従うと、支払人は銀行券本体を送らない限り、代金の支払いを証明するための確認情報は得られない。しかし、受取人は銀行券本体を受取った後に確認情報を送るので、銀行券本体を受け取っていないふりをすることで、代金の支払いを受けていないふりをすることができる。実際には受取人は「正しい」銀行券本体を手に入れているので、他者へ「裏書譲渡」することが可能となる。この「正しい」銀行券は、「検証」の際に不正使用として扱われることはない。もし、もとの銀行券を支払人が別の受取人に「裏書譲渡」した場合には、「検証」によって支払人の複製による不正使用とみなされてしまう。これを防ぐためには、銀行券本体を送る前に「裏書譲渡」を行っていた証拠を得ておく必要がある。

そこで、銀行券本体を送る前に、受取人から「裏書譲渡」を行っていた証拠を得ることとした。つまり、あらかじめ受取人から確認情報（領収書）をもらい、それから支払人が銀行券本体を送信するようにするのである。ただし、「領収書」は再利用できないようにする。そして、銀行券本体到達後、受取人は受取確認情報を送る。

「領収書」は、再利用できないようにセッション ID を含み、かつ証拠能力があるように銀行券ごとに違うことが保証されている情報（第 3.3. 1節）、すなわち銀行券番号を含む。実際には、受取人はその銀行券番号が現在自分が「裏書譲渡」を受けようとする銀行券のものかどうかわからないので、ID と銀行券番号の組である必要がある。ID と銀行券番号には、支払人や受取人が勝手に捏造できないように発行局の署名が施されており、これとセッション ID に対して、受取人の署名が施される。このようにすることで、「領収書」はある銀行券をあるセッション ID を使って取引したことを受取人が保証する情報となる。この情報はセッション毎に違い、支払人だけでは絶対に構成不可能な情報である。

$$\text{領収書}：((ID, \text{銀行券番号})^{SK_{\text{発行局}}}, (\text{セッション } ID))^{SK_{\text{受取人}}}$$

この方法でも、受取確認情報が送られてこない可能性はある。「領収書」を得たにもかかわらず、支払人が銀行券本体を送らなかった場合と、銀行券本体を受け取ったにもかかわらず、受取人が受取確認情報を送らなかった場合である。どちらの場合についても、互いに送ったもしくは受け取っていないと主張しあった時には、どちらの主張が正しいのか

を確かめるのは難しい。しかし、「領収書」があるので、この方法の場合、その銀行券を使って「裏書譲渡」のトランザクションをしていたという事実がある。

そこで、このような場合は支払人がそのセッションで送ったはずの銀行券本体をもう一度再送することにする。そのためには、受取確認情報を得られなかった場合、そのセッションで送った銀行券本体を支払人は保存する。そして、何らかの手段によって銀行券本体が到達していないことが判明した場合、受取人に対して保存してある銀行券本体を再送する。この再送の導入により、確実に銀行券本体が届くことが期待できる。そのため、「領収書」がある場合はすでに「裏書譲渡」されたものとしてその銀行券は扱う必要がある。支払人は、別の受取人に「裏書譲渡」しないようにしなければならない。また、「検証」も行ってはならない。「裏書譲渡」や「検証」をすることは複製使用に値するからである。再送は、銀行券が届いていないときのための補完機能であり、銀行券本体の複製を送るだけなので、受取人がそれらを再送を受けた回数分だけ使用すれば、複製使用となる。つまり、受取人が再送によって不当な利益を受けることはない。

このように、「領収書」と「銀行券本体の再送」を導入することで、代金到達について問題が発生した場合に対処することができる。しかし、第 3.2.1 節の想定に従うと、通信路において盗難がありうるため、たとえ銀行券本体や確認情報を送ったとしても相手に届いているという保証はない。そこで前述の方法では、何らか別の方法によって確認する必要がある。このような、情報の「到達の確認」に関する議論は第 8.1 節で詳しく行うこととする。

その他の留意点　その他の留意点は、以下のとおり。支払人／受取人がセッション毎に生成する乱数は、セッション毎に違うことが期待される乱数だが、正確には「ある支払人が同じ受取人と同じ銀行券を取引する場合」に対して、唯一の値が与えられる程度には充分な強度のある乱数であるとする。つまり、ある人が自分の関わったあるセッションを特定するためにはセッション ID で充分であるが、発行局や裁判所があるセッションを特定するためには領収書が必要となるということである。

トランザクションの概略

「裏書譲渡」のトランザクションの概略は以下のとおり。

署名したセッション ID（裏書譲渡するたびに毎回違う情報）を交換し、受取人は「領収書」を支払人に送り、銀行券の裏書情報として、互いに相手の署名したセッション ID

に自分の署名を施したもの（その相手とそのセッション ID で取引したことを証明する情報）を書き込む。これにより、相手が自分と取引したことを否定できないようにする。受取人は最後に確認情報として、受け取った銀行券の額面情報とセッション ID に自分の署名を施して送る。

確認情報が送られてこなかった場合、支払人は「領収書」によって、支払いを証明することができる。ただし、支払人は受取人からの代金の請求があった場合には、保存してある銀行券本体を再送する義務がある。

トランザクションの詳細

1. 公開鍵証明の交換を行なう。
2. 受取人は乱数を生成し、署名し（受取人にしか作れず、毎回違う情報）、暗号化して（支払人しかその内容を知ることはできない）送る。

$$\text{支払人} \leftarrow \text{受取人} : ((\text{受取人の乱数})^{SK_{\text{受取人}}})^{PK_{\text{支払人}}}$$

3. 支払人は暗号を解き、署名を検証する。
4. 乱数を生成し、署名し（支払人にしか作れず、毎回違う情報）、ID と受取人の書名がついた受取人の乱数とともに署名し（受取人の情報をもとに作る、ある銀行券と ID によって結びついた支払人にしか作れない毎回違う情報。セッション ID）、額面情報の中から取り出した金額に発行局の署名が施されたもの（発行局にしか作れない情報）とともに暗号化して（受取人しかその内容を知ることはできない）送る。

$$\text{支払人} \rightarrow \text{受取人} : ((\text{セッション } ID)^{SK_{\text{支払人}}}, (\text{金額})^{SK_{\text{発行局}}})^{PK_{\text{受取人}}}$$

5. 受取人は暗号を解き、署名を検証し、金額と乱数を確認する。
6. セッション ID に署名し（支払人の情報をもとに作る、受取人にしか作れない毎回違う情報）、暗号化して（支払人にしかその内容を知ることはできない）送る。

$$\text{支払人} \leftarrow \text{受取人} : ((\text{セッション } ID)^{SK_{\text{受取人}}})^{PK_{\text{支払人}}}$$

7. 支払人は暗号を解き、署名を検証し、セッション ID を確認する。

ここまでで、支払人と受取人にはそれぞれ相手の署名がなされたセッション ID を入手している。

8. 額面情報の中から取り出した ID と銀行券番号に発行局の署名が施されたもの（発行局にしか作れない情報）を暗号化して（受取人しかその内容を知ることはできない）送る。

$$\text{支払人} \rightarrow \text{受取人} : ((ID, \text{銀行券番号})^{SK_{\text{発行局}}})^{PK_{\text{受取人}}}$$

9. 受取人は暗号を解き、署名を検証し、ID を確認する。

10. ID と銀行券番号に発行局の署名が施されたものとセッション ID に署名し（支払人の情報をもとに作る、受取人にしか作れない毎回違う情報。領収書）、支払人の署名が施されたセッション ID に署名し（そのセッション ID で取引したことを支払人が証明したことを受取人が証明した情報）とともに暗号化して（支払人にしかその内容を知ることはできない）送る。

$$\text{支払人} \leftarrow \text{受取人} : (((ID, \text{銀行券番号})^{SK_{\text{発行局}}}, \text{セッション } ID)^{SK_{\text{受取人}}},$$
$$((\text{セッション } ID)^{SK_{\text{支払人}}})^{SK_{\text{受取人}}})^{PK_{\text{支払人}}}$$

11. 支払人は暗号を解き、署名を検証し、領収書とセッション ID を確認する。

12. 自分の署名が施されたセッション ID に受取人が署名を施したものを裏書情報に加えて支払人の ID と受取人の ID（後で検証できるように）とも発行局の公開鍵で暗号化し（受取人からの情報と裏書情報をもとに作る発行局のみが内容を検証できる情報。新しい裏書情報）、受取人の署名がなされたセッション ID に署名をし（受取人からの情報を基に作る支払人にしか作れない情報）、額面情報とともに暗号化して（使用することができるのは受取人のみ）送る。

$$\text{支払人} \rightarrow \text{受取人} : ((\text{裏書情報}, ((\text{セッション } ID)^{SK_{\text{支払人}}})^{SK_{\text{受取人}}},$$
$$\text{支払人の } ID, \text{受取人の } ID)^{PK_{\text{発行局}}},$$
$$((\text{セッション } ID)^{SK_{\text{受取人}}})^{SK_{\text{支払人}}}, \text{額面情報})^{PK_{\text{受取人}}}$$

13. 受取人は暗号を解き、署名を検証し、セッション ID と額面情報を確認する。

14. 自分の署名が施されたセッション ID に支払人が署名を施したものを受けとった情報の裏書情報部分（発行局の公開鍵で暗号化されている部分）に加えて支払人の ID と受取人の ID（後で検証できるように）とともに発行局の公開鍵で暗号化して（支払人からの情報と裏書情報をもとに作る発行局のみが内容を検証できる情報。新しい裏書情報）、額面情報を加え、銀行券実体として保存する。

$$\begin{aligned}&\text{受取人} : (((\text{裏書情報}, ((\text{セッション } ID)^{SK_{\text{支払人}}})^{SK_{\text{受取人}}},\\ &\text{支払人の } ID, \text{受取人の } ID)^{PK_{\text{発行局}}},\\ &((\text{セッション } ID)^{SK_{\text{受取人}}})^{SK_{\text{支払人}}},\\ &\text{支払人の } ID, \text{受取人の } ID)^{PK_{\text{発行局}}}, \text{額面情報})\end{aligned}$$

15. 受取人が確実に銀行券を受けとったことを知らせるために、額面情報とセッション ID に署名し（額面情報を基に作る受取人にしか作れない情報）、暗号化して（支払人のみがその内容を知ることができる）送る。

$$\text{支払人} \leftarrow \text{受取人} : ((\text{額面情報}, \text{セッション } ID)^{SK_{\text{受取人}}})^{PK_{\text{支払人}}}$$

16. 支払人は暗号を解き、署名を検証し、額面情報を確認する。保存してあった銀行券実体を削除する。

3.3.3 検証・分割

銀行券の検証を要求する人を「検証人」とし、実際の検証を行うものを「発行局」とする。

トランザクションにおける留意点

検証が済んだ銀行券は、不正がなかった場合にはそのままで、あった場合には遡及によってその損失が補填されることで、額面どおりの価値を持つと考えられる。そのため、発行局は受取った銀行券と引換えに同じ額面の新しい銀行券を振り出すことができる。

本方式では、「裏書譲渡」の際に、裏書情報として署名の連鎖が銀行券に記録されていく。そのため、銀行券本体は「裏書譲渡」の度に大きくなっていくという問題がある。「検証」にはこの問題を解決する機能も付加することとした。すなわち、上述のように検証の後に同じ額面の新しい銀行券を振り出すことで、銀行券の裏書情報を圧縮するのである。

なお、「分割」はこの「検証」を拡張したものに過ぎない。検証が済んだ銀行券が額面どおりの価値を持つなら、それを分割して振り出してもらっても問題がない。そこで、「分割」は「検証」同様に、新しい銀行券を振り出してもらうが、任意の額面に分割して振り出してもらうこととした。この考えを応用すれば「結合」も可能である。

トランザクションの概略

「検証」および「分割」のトランザクションの概略は以下のとおり。

検証人は署名した金額（「分割」の場合は複数）と銀行券本体を送り、発行局では受取った額面情報と裏書情報を検証し、不正が検出された場合には、不正者を特定して遡及する。検証後は同じ額面の新しい銀行券が振り出されるため、「振出」のトランザクションに準ずる。

検証人自身が不正を行っていた場合には、新しい銀行券が振り出されることはない。

トランザクションの詳細

1. 公開鍵証明の交換を行なう。
2. 検証人は検証しようとする銀行券の金額（「分割」の場合は複数）に署名をし（検証人にしか作れない情報）、検証しようとする銀行券本体とともに暗号化して（発行局しかその内容を知ることはできない）送る。

$$\text{検証人} \rightarrow \text{発行局} : ((\text{金額})^{SK_{\text{検証人}}}, \text{銀行券本体}))^{PK_{\text{発行局}}}$$

3. 発行局は暗号化を解き、署名を検証し、銀行券の検証を署名を基に行なう。検証が終了したら、銀行券を使用済みとして保存しておく（後で複製使用を検出するため）。
4. 検証が正常に終ったら検証人を振出人とする同じ額面の新しい銀行券にするために、振出トランザクション（第 3. 3. 1節）の 4 から始める。「分割」の場合は、分割数分繰り返す。

3.3. 4　引換

銀行券の引換を要求する人を「引換人」とし、実際の引換を行うものを「発行局」とする。

トランザクションにおける留意点

「引換」は「検証」と原理的に全く同じである。ただし、「引換」では同じ額の新しい銀行券を発行する代わりに、現金化（口座振り替えなど）を行う。そのため、トランザクションは「検証」より簡単で、新しい銀行券が振り出される代わりに、引換ができたかどうかが引換人に通知される。

トランザクションの概略

「引換」のトランザクションの概略は以下のとおり。

引換人は署名した金額（「分割」の場合は複数）と銀行券本体を送り、発行局では受取った額面情報と裏書情報を検証し、不正が検出された場合には、不正者を特定して遡及する。検証後は現金化が行われ、引換が行われたことが引換人に通知される。

引換人自身が不正を行っていた場合には、現金化は行われず、引換ができなかったことが引換人に通知される。

トランザクションの詳細

1. 公開鍵証明の交換を行なう。

2. 引換人は引換しようとする銀行券の金額に署名をし（引換人にしか作れない情報）、引換しようとする銀行券本体とともに暗号化して（発行局しかその内容を知ることはできない）送る。

$$\text{引換人} \rightarrow \text{発行局} : ((\text{金額})^{SK_{\text{引換人}}}, \text{銀行券本体}))^{PK_{\text{発行局}}}$$

3. 発行局は暗号化を解き、署名を検証し、銀行券の検証を署名を基に行なう。検証が終了したら、銀行券を使用済みとして保存しておく（後で複製使用を検出するため）。

4. 引換が終ったらその結果を引換人に通知する。額面情報と” 正否” に署名し（発行局にしか作れない銀行券毎に違う情報）、暗号化して（内容を知ることができるのは引換人のみ）送る。

$$\text{引換人} \leftarrow \text{発行局} : ((\text{額面情報},'' \text{正否}'')^{SK_{\text{発行局}}})^{PK_{\text{引換人}}}$$

第 4 章

実装

本方式の実験、評価および実現のために、実装を行った。本章では、実装方法と詳細、現状について報告する。

4.1 実装方法

実装は、BSD/OS Ver 2.1 上で行った。実装したシステムは、発行局側のサーバ／クライアント、利用者側のサーバ／クライアントの四つである（表 4.1 ）。今回、公証局は発行局側のサーバに含むこととし、別のシステムとして分離しなかった。

発行局側のサーバ	銀行券の振出、検証、引換をする、公開鍵証明書の発行
発行局側のクライアント	銀行券 DB、口座 DB、公開鍵 DB の保守
利用者側のサーバ	銀行券の裏書譲渡を受ける
利用者側のクライアント	銀行券の振出、検証、引換を受ける、裏書譲渡する、 銀行券 DB、公開鍵 DB の保守

表 4. 1: 実装システムとその役割

これらのシステムはすべて、メッセージの内容の暗号化／復号化や署名／署名検証を行う暗号化モジュールと、メッセージを生成および解釈し、送受信する通信モジュールとユーザーインターフェイスから構成されている。今回は、提案方式の電子決済システムとしての動作を実現することに主眼をおいた。そのため、ユーザーインターフェイスは簡易

なものにとどめ、暗号系や通信についてもなるべく既存のものを利用して、実装負担を主眼点に集中させた。

以降に、暗号化モジュールが用いる暗号系の選択についてや、通信モジュールが実際に通信するメッセージについてを述べ、実装の現状を報告する。

4.2 暗号系の選択

暗号化モジュールは、メッセージの内容について、公開鍵による暗号化／秘密鍵による復号化および、秘密鍵による電子署名／公開鍵による署名検証、慣用鍵による暗号化／復号化を行なうモジュールである。よって、非対称鍵暗号系と慣用鍵暗号系が必要となる。以下に、簡単に各暗号方式を概観し、選択理由を述べる。

4.2.1 公開鍵暗号／電子署名方式の選択

非対称鍵暗号系（公開鍵暗号系）には DH 鍵交換方式 [5]、楕円 DH、RSA 公開鍵暗号 [12]などがある。暗号化モジュールは可換なので、どれを用いても構わないが、

- 電子署名が提供されているか
- 実装例が存在するか

などの観点から、今回の実装において使用するものを決定することとした。

DH 鍵交換方式

DH[5]単体だけでは電子署名が提供されておらず、DH は鍵交換方式なので他の暗号化方式と組み合わせる必要がある。また、DH の実装例は残念ながら見つからなかったので、今回は使用しないこととした。

楕円 DH

楕円 DH は、DH を楕円方程式に基づいて拡張したものであり、電子署名や暗号化も提供されている。楕円 DH の実装については、手に入らないので、今回は見送りとした。

RSA 公開鍵暗号

RSA 公開鍵暗号 [12] は、公開鍵暗号系でも特に有名なもので、電子署名や暗号化が提供されている。ソースについては、RSAREF という非商用使用目的については無料のパッケージが存在するが、アメリカ国外ではこれを用いることはできない。しかし、RSA公開鍵暗号を採用していて、かつ自由配布のソフトウェアである PGP のソースには RSA公開鍵暗号に関する実装例が含まれている。このうち、pgp2.6ui 以降のいわゆるインターナショナル版は、アメリカ国外でも使用可能である。そこで、今回は pgp2.6.3のソースをベースに暗号化モジュールを作成することとした。ただし、RSA公開鍵暗号は複雑であるため、処理時間がかかるので、高速な慣用鍵暗号と組み合わせて用いることとする。

4.2.2 慣用鍵暗号方式の選択

対称鍵暗号系（慣用鍵暗号系）には DES[9] 多段 DES、RC シリーズ [14] 、IDEA 、その他多くの方式がある。

今回は公開鍵暗号方式に RSA公開鍵暗号を用い、pgp2.6.3のソースをベースとすることとしたので、その内部に IDEA 暗号化／復号化モジュールが含まれていたのでそれをそのまま用いることとした。

DES、多段 DES

DES (DataEncryption Standard)は、[9]鍵長 56 bit のブロック型暗号で、アメリカの政府標準暗号方式である。既に DES については多くの攻撃法が知られており、安全とはいえない。そこで、DES を多段化し、鍵長を伸ばすのが多段 DES である。しかし、この多段 DES では、当然暗号化／復号化が遅くなる上、最近では様々な攻撃法が提案されており、安全性には疑問の声も上がっている。

RC シリーズ

RC(Rivest Cipher)シリーズ [14]には、RC2（可変長鍵、ブロック型暗号）、RC4（可変長鍵、ストリーム型暗号）、RC5（鍵長、ブロック長、ラウンド数の全てが可変）があり、このうち RC5 のみアルゴリズムが公開されている。この方式は 1995 年に発表されたばかりで、非常に新しい方式であり、現在基本的な安全性について検証が行なわれている

最中である。ある程度安全性が検証されれば、このアルゴリズムは非常に単純なので、高速な処理が可能であり、画期的な暗号方式といえる。

IDEA

IDEA(International Data Encryption Agorithm) は、鍵長 128bit のブロック型暗号で、比較的新しい暗号である。今のところ欠陥はみつかっておらず、また、アメリカ以外の国での使用に全く制限はない。アメリカで実装された場合はアメリカ国外への持ち出しが禁止される（これはある程度以上強い暗号全てに適用される)。

4.3 メッセージの構造と内容

通信モジュールが生成および解釈するメッセージの構造とその内容について述べる。メッセージの一般的な構造については、RFC1991[13] や PKCS #7[15]を参考に決定した(図 4.1)。

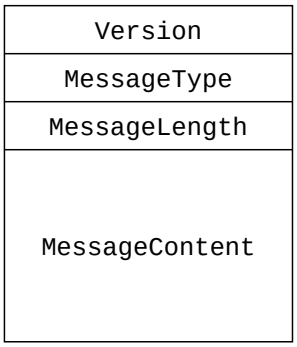


図 4.1: Messageの一般的構造

メッセージは一般的に、システムのバージョン（Version) とそのメッセージの型（Me s s aygpe、T表 4.2)、メッセージの長さ（MessageLength）とメッセージ本体（MessageCon t ent）から成る。メッセージ本体は、いくつかのコンテントからなり（図 4.2)、コンテント自身もいくつかのコンテントを内包しうる。

コンテントは一般的に、コンテントの型（Cont e tnTy pe、表 4.3 ）とシステムのバージョン（Version)、コンテント本体の長さ（ContentLength）を含む、コンテントヘッダ

鍵交換	KEY_EXCHANGE_REQ/REP/ ACK/ NAK
振出	ISSUE_REQ/ CHA/ RES / REP/ ACK/ NAK
裏書譲渡	ENDORSE_CHA/ RES / REQ/ REP/ REC/ CHE/ ACK/ NAK
検証／引換	CHECK_REQ/ ACK/ NAK

表 4.2: MessageType 一覧

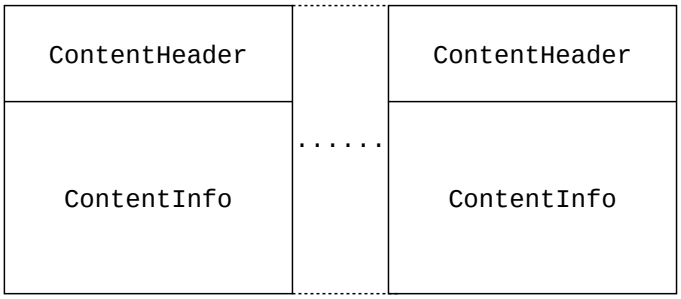


図 4.2: MessageContent の構造

とコンテント本体（ContentInfo）から成る。コンテントヘッダの長さは、各コンテント型毎に固定である。

各型のコンテントと各メッセージの詳細について、以下に示す（図 4.3）。

4.3.1 Data

Data型は生データである。つまり、全く加工されていない生のデータで、ほとんどの型のコンテントの本体はこの Data型のコンテントを加工したものである。Data型のヘッダは、コンテントの型（ContentType）、システムのバージョン（Version）、コンテント本体の長さ（ContentLength）から成る。コンテント本体は、生データの Octet-Stream である。

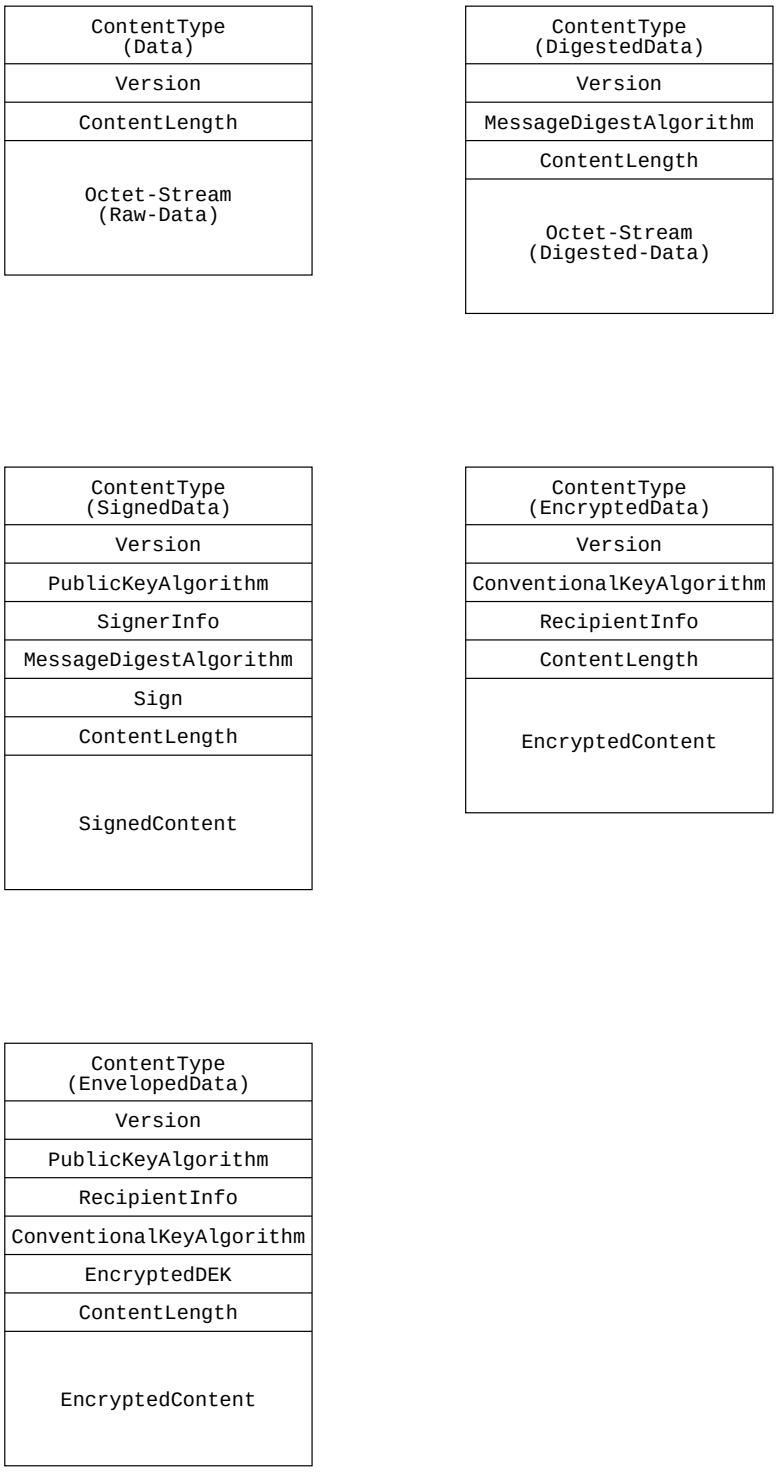


図 4.3: 各 Content の構造

Data	生データ
SignedData	署名付きのデータ
EnvelopedData	公開鍵で暗号化済みのデータ
DigestedData	データのダイジェスト
EncryptedData	慣用鍵で暗号化済みのデータ

表 4.3: ContentType 一覧

4.3.2 SignedData

SignedData型は署名付きのデータである。コンテント本体のメッセージ・ダイジェストに対し、秘密鍵による暗号化を施したもの、すなわち、電子署名がコンテントヘッダに含まれる。SignedData型のヘッダは、コンテントの型（ContentType）、システムのバージョン（Version）、公開鍵暗号のアルゴリズム（PublicKeyAlgorithm）、署名者の情報（SignerInfo）、メッセージ・ダイジェストのアルゴリズム（MessageDigestAlgorithm）、電子署名（Sign）、コンテント本体の長さ（ContentLength）から成る。コンテント本体は、他のコンテントが並んだものである。

4.3.3 EnvelopedData

EnvelopedData型は公開鍵で暗号化済みのデータである。実際には、公開鍵暗号系は複雑で処理が重いために、セッション毎に違う慣用鍵（DEK: DataEncryptKey）を作成し、コンテント本体はDEKで暗号化し、DEKを公開鍵で暗号化してヘッダ内に書き込んでおく。EnvelopedData型のヘッダは、コンテントの型（ContentType）、システムのバージョン（Version）、公開鍵暗号のアルゴリズム（PublicKeyAlgorithm）、受信者の情報（RecipientInfo）、慣用鍵暗号のアルゴリズム（ConventionalKeyAlgorithm）、公開鍵で暗号化済みのDEK（EncryptedDEK）、コンテント本体の長さ（ContentLength）から成る。コンテント本体は、本来のコンテント本体をDEKで暗号化したものである。

4.3.4 DigestedData

DigestedData 型はデータのメッセージ・ダイジェストである。すなわち、データ自身は含まない。DigestedData 型のヘッダは、コンテントの型（ContentType）、システムのバージョン（Version）、メッセージ・ダイジェストのアルゴリズム（MessageDigestAlgorithm）、コンテント本体の長さ（ContentLength）から成る。コンテント本体は、他のコンテントが並んだもののメッセージ・ダイジェストである。

4.3.5 EncryptedData

EncryptedData 型は慣用鍵で暗号化済みのデータである。EncryptedData 型のヘッダは、コンテントの型（ContentType）、システムのバージョン（Version）、慣用鍵暗号のアルゴリズム（ConventionalKeyAlgor）、受信者の情報（RecipientInfo）、コンテント本体の長さ（ContentLength）から成る。コンテント本体は、他のコンテントが並んだものを慣用鍵で暗号化したものである。

4.3.6 各メッセージの詳細について

各メッセージの書式は、付録 A の図 A.1 〜図 A.14 に示す。図中の各ブロックは項目をあらわし、括弧内は本実装におけるその項目に入るべき数値や定数である。なお、"Issuer" は発行局、"Endorser" は支払人、"Endorsee" は受取人、"User" は振出人や検証人、引換人、"EndorseInfo" は裏書情報、"FaceValueInfo" は額面情報、"BankNote" は銀行券本体を示す。

各メッセージはこの書式にしたがって、生成、送信され、受信、解釈される。解釈の際に各メッセージは、書式にしたがっているかどうか検査される。書式にしたがっているかどうかは、まずコンテントヘッダから判断する。コンテントの型、システムのバージョン、各種のアルゴリズム、受信者／署名者の情報を検査し、正しかった場合、コンテント本体を検査するために、コンテントの型毎に復号化や署名の検証処理などを行なう。すべてが正しかった場合、コンテント本体の中身を検査する。

裏書情報は、発行局が「検証」の際に暗号鍵を特定するために支払人や受取人の ID を含むことになっているが、本実装は、読み取り可能なコンテントヘッダによって暗号鍵を特定することができるので、利用者の ID をコンテントとして別途格納する必要はない。

4.3.7 メッセージのやり取り

各トランザクションにおけるメッセージの正常時のやり取りを、タイムチャートで図 4.4 に示す。左から順に、「鍵交換」、「振出」、「裏書譲渡」、「検証／引換」である。”User Client” は利用者側のクライアント、”User Server” は利用者側のサーバ、”Bank Server” は発行局側のサーバを示す。

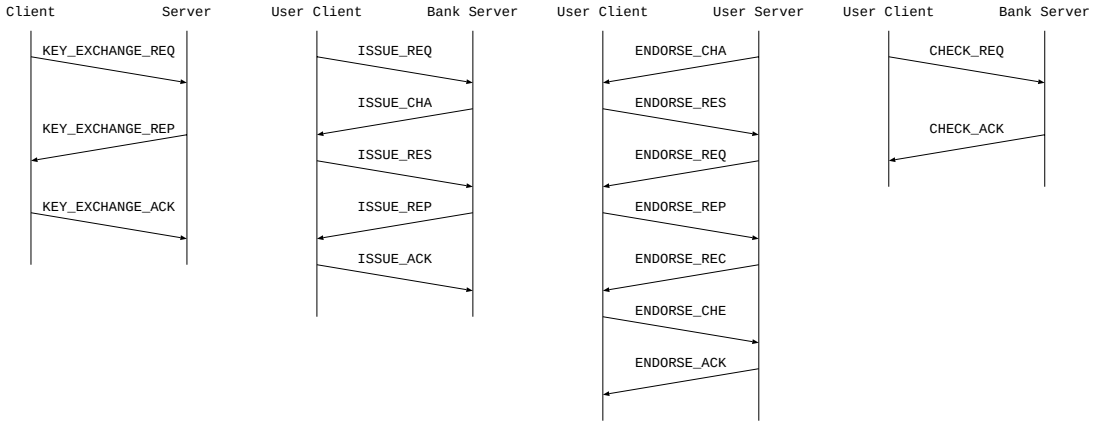


図 4. 4: メッセージのやり取り

4.4 実装状況

実装は、現在αテスト段階である。いくつかのバグや不完全な機能が存在する。また、サンプル実装の中には、いくつか実装されていない機能がある。例えば、「分割」は実装されていない。他にも細かな点で実装されていない機能がある。また、ユーザーインターフェイスは暫定的なものなので、別途設計／実装を行い、切り替えていく予定である。

本実装におけるメッセージはメッセージの型やコンテントヘッダの一部が露出しているため、トラフィック解析やその他の攻撃に対する安全性が弱いと思われる。暗号化部分についても、一般に公開されている公証局を利用可能にするために、RFC199 1 [1 3]や PKCS #7[1 5]に対応するなどの改善が必要である。また、今回は暗号系の安全性より実装例の存在等を重視し、公開鍵暗号系として RS A公開鍵暗号を、慣用鍵暗号系として IDEA を採用したが、実際に運用可能なものとするためには、暗号系自身の安全性についても充分な検討を行なう必要があると考えている。

また、現在サーバは単純な反復サーバとして実装されているが、スケーラビリティを考慮すると、並行サーバにすることや機能分割して暗号化／復号化などの処理を別の機械で実行することなどが必要となってくると考えられる。鍵や銀行券や口座の DB も、現在は BSD/OS のファイルシステムをそのまま利用した簡易的なものなので、大規模運用ができるように再設計が必要となると考えている。

このように、安全性やスケーラビリティを確保し、一般に普及可能なものにするためには、全体を再設計し、実装する必要がある。

第 5 章

裏書譲渡可能な銀行券方式の課題

本方式において、いくつかの解決困難な問題点がすでに指摘されている。解決法について考察したので、問題点を整理した上で述べる。

5.1 匿名性の提供

本方式は、発行局が不正を事後検出し、不正者に対し遡及するために、発行局は銀行券の流通経路をすべて知ってしまう。すなわち、本方式では発行局に対する匿名性が提供できないという欠点がある。発行局がその情報を不正の検出以外に利用しないことが保証できれば問題ないが、発行局が一般的には銀行などであることを考えると難しい。そこで、この問題は、通貨の重要な特性を提供できないという意味では、大きな欠点であるといえる。

そこで、発行局に対する匿名性を提供するための拡張として、銀行券の取引を仮名によって行うことを考え、そのために仮名を管理する機構（図 5.1 ）を導入する。この仮名を管理する機構のアイデアは、[1] の中で提案されている利用者のプライバシー保護のための機構”Trustee” に基づいている。

この仮名を管理する機構は、発行局と結託することがなく、仮名に関する情報を第三者に漏らすことがない、信用のある独立した第三者機関であるとする。これを仮に、仮名管理機関と呼ぶこととする。

銀行券の取引を行う利用者は、仮名管理機関に登録し (a)、仮名の発行を受ける (b)。仮名管理機関はこの際に、実名と仮名の組み合わせを記録する。また、銀行券の「振出」や

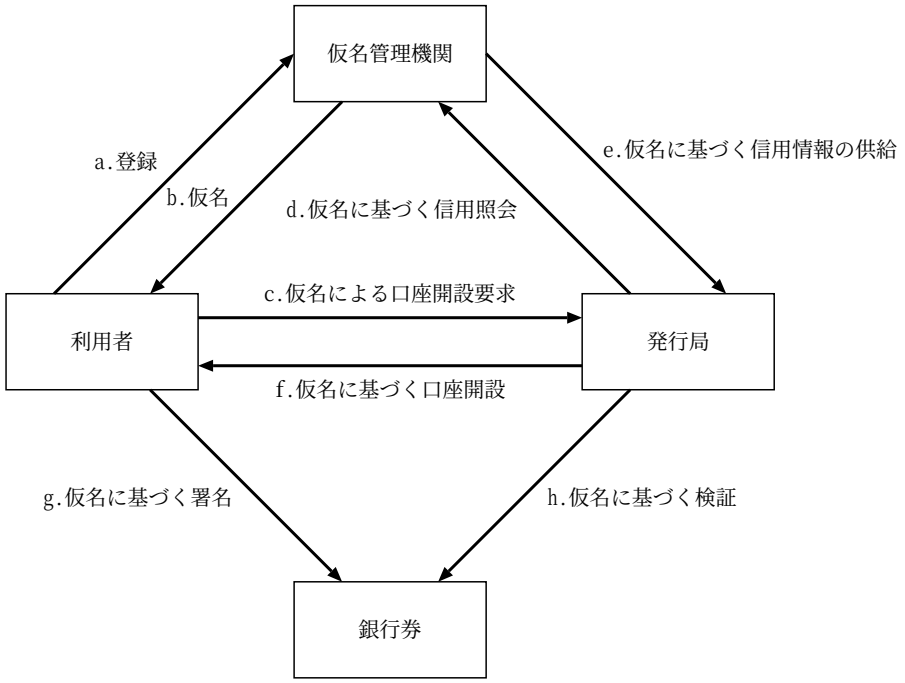


図 5.1: 仮名による取引の概念

「引換」を受ける利用者は、この仮名に基づき発行局に口座の開設を依頼する (c)。発行局はこの際に、仮名管理機関に仮名に基づく信用情報の照会を依頼し (d)、仮名管理機関は、仮名に基づいて信用情報を提示する (e)。発行局はこの信用情報から、仮名に基づく口座を開設する (f)。銀行券の取引を行う際には、利用者は仮名に基づく電子署名を用いる (g)。発行局は銀行券を仮名に基づき検証する (h)。「検証」の際に不正を発見した場合、仮名管理機関に不正者の仮名と不正の内容を通知する。仮名管理機関はこの不正の通知に基づき、仮名から実名を導き、不正者に対し責任を遡及する。

このような機構を導入すると、発行局は仮名による流通経路を知ることはできるが、実名を伴う実際の流通経路を知ることはできない。また、仮名管理機関は不正の通知を受けたときに、不正者と不正の内容を知ることはできるが、それ以外の銀行券の流通については何も知ることはできない。よって、発行局に対する匿名性を提供することができる。

しかし、この方法にはいくつかの制約や欠点がある。

一つは、利用者は仮名に基づく電子署名を行うために専用の鍵を作成しなければならず、仮名に基づく鍵の証明を行う専用の公証局も必要となる。これは、すでに開設されて

いる公証局を利用できないという意味から、普及力の面で制約となる。

もう一つは、仮名による口座の開設は、発行局が一般的に銀行のような金融機関であることを考えると、現行の法制度では難しい。現在の金融機関は決済機関と信用機関の両方の側面を持っているが、信用情報を仮名管理機関が提供する場合、金融機関は決済機関としての役割のみしか持たないこととなり、現在の制度上では実現が難しいと考えられる。

5.2 社会的信用の維持

不正を事後検出する本方式のような電子通貨においては、複製などによる不正使用が行われてから、「検証」によってそれが発覚するまでに時間差が生ずる。この不正使用と「検証」の間の時間差を利用して、大量の複製を行い、不正使用した場合、短時間に複製が大量に流通することになる。これによって、多額の実体を伴わない通貨が流通する事となり、通貨の与信力は低下する。このような通貨の与信力の低下が頻繁に起こると、通貨の社会的信用は低下する。

この不正による被害は、遡及によって不正者から補填されることが期待されるが、この種の不正は不正者の経済能力に関わりなく行う事ができるので、不正者が破産した場合などを考えると無視できないほどのリスクが発生する。不正使用と「検証」の間の時間差が大きいほど、このリスクは増大する。このリスクが大きくなれば大きくなるほど、実質的な通貨の与信力は低くなるので、通貨の社会的信用は低くなると考えられる。そのため、本方式の提供する銀行券の社会的信用は「検証」の頻度に依存するといえる。

このリスクは、複製された銀行券の発行局と受取人が負うことになる。そのため、このリスクの制御は発行局と受取人の側で行える必要がある。本方式においては、利用者の利便性と受取人によるリスクの制御を考え、「検証」は利用者が任意の時点に行う事ができることになっている。受取人は、自分が「検証」することによってリスクを制御することができる。しかし、発行局は自分自身が「検証」することはできないので、リスクを制御することはできない。本方式に何ら拡張を加えなければ、「検証」の頻度は利用者の本方式への信用に依存し、それは必ずしも発行局の意図とは一致しない。そこで、このリスクを制御するための何らかの手段が発行局に与えられなければ、発行局が本方式を受け入れるのは難しいと思われる。

そこで、本方式の社会的信用を維持し、発行局がリスクを制御するために、発行局による「検証」頻度の制御を可能とする拡張を考える。

本方式では、発行局が常に銀行券の流通に関わるわけではない。発行局が関わることができるのは、「振出」、「検証」、「引換」のときのみである。そのため、発行局が「検証」の頻度を制御するのであれば、「振出」のときに何らかの仕掛けを銀行券に施す以外に方法はない。そこで、銀行券に対し、「振出」の際に何らかの期限をつけることを考える。

この期限は、期限内に「検証」することを促すもので、発行局が決定し、「振出」の際に銀行券に参照可能で書き換え不可能な情報として付加する。期限が切れた銀行券において「検証」によって不正が発見され、不正者が支払能力を持たなかった場合、その損失については期限が切れたにもかかわらず「検証」を行わなかった受取人が負うとする。

期限が設定された銀行券を受け取った受取人は、必ずしも期限内に「検証」しなくてもかまわない。期限が切れた銀行券は、「検証」において不正が発見されなければ何ら問題とならないからである。しかし、「検証」において不正が発見され、不正者が支払能力を持たなかった場合はその損失を負わねばならない。そのため、受取人は自分のリスクを制御するために、銀行券に対する信用と期限との兼ね合いから「検証」の時期を決定すると考えられる。

このような仕組みを導入することによって、期限によって発行局は「検証」の頻度を緩やかながら制御することができる。また、期限によってより明確に受取人に対してリスクを配分することができるので、受取人はより銀行券の信用に注意を払うようになると考えられる。よって本方式の提供する銀行券は、期限の導入により、発行局と受取人にとって必要充分な社会的信用の維持が待期できる。

しかし、この方法にはいくつかの制約や欠点がある。

一つは、期限の実現法に伴う問題である。期限の実現法としては、時間で設定する方法や「裏書譲渡」の回数で設定する方法が考えられる。期限を時間で設定した場合、時計の同期の問題が起こる。時計が同期していなければ期限が切れる時間が発行局と受取人の間で異なり、受取人は期限内に「検証」したつもりが、発行局では期限切れの「検証」として扱われる可能性がある。つまり、期限を時間で設定する場合には、発行局と利用者の間で時計を同期するための何らかの仕組みが必要となる。また、期限を「裏書譲渡」の回数で設定した場合、「裏書譲渡」回数の管理の問題が起こる。回数を利用者が書き込む場合、期限が切れているにもかかわらず期限が切れていないように見せかける不正が可能となる。これを防ぐために、回数を発行局が管理しようとした場合、「裏書譲渡」の際には必ず発行局が介在しなければならなくなってしまう。つまり、期限を「裏書譲渡」の回

数で設定する場合には、「裏書譲渡」の回数を管理するための何らかの仕組みが必要となる。以上のように、期限の実現のためには何らか新しい仕組みを考え出し、導入する必要があるという問題がある。

もう一つは、期限のついた銀行券の信用の問題である。受取人は、信用の持てない支払人からの期限の切れた銀行券の受取りは拒否すると考えられる。では、期限が切れる直前の銀行券はどうだろうか。これも、なるべく受取りたくないと考えるであろう。つまり、銀行券の信用は期限が近づくにつれて低くなると考えることができる。銀行券の信用が期限が近づくにつれて低くなるため、受取人は期限が近くなるにつれて、「裏書譲渡」ができなくなる。そのため、「検証」によって期限を再設定してもらう必要がある。これが極限まで進むと、「裏書譲渡」するためには直前に必ず「検証」することが考えられる。これは、発行局側でも言えることで、損失の可能性を極限まで減らすためには「検証」を「裏書譲渡」の度にせねばならないように期限を設定することが考えられる。つまり、期限の導入により、社会的信用は維持できるが、その維持のために、発行局が介在することなく転々譲渡が可能であるという本方式の優位点が失われてしまう可能性がある。

第 6 章

比較評価

インターネット上での決済システムについては、近年多くの研究が行われており、既にいくつか実現され、実際に決済に利用されているものもある。そこでその内のいくつかを取り上げ、本方式と比較することにより、評価を行うこととする。

6.1 関連研究

比較対象として実際に取り上げるのは、第 2.4.1 節に示した各メカニズム毎に実現されているものを一つないしは二つ、それ以外にインターネット上のシステムではない実験段階のものを一つと、提案中のものを一つ、合計六つである。

取り上げるものは以下のとおり。安全なクレジットカード番号の提示法として、FIRST VIRTUAL[18] と CyberCash[16] 電子小切手システムとしては、NetCheque[22]。電子通貨システムとして、ecash[17]。その他のものとして、現在イギリスのスウィンドン市などで実験が行われている電子通貨 MONDEX[20] と最近提案された供託電子マネー [8] [11] [1] (単一の方式ではないが、匿名性を保ちながら不正を事後検出できるため今後の電子通貨方式の主流となると思われる)。

それぞれの方式を概観し、比較の観点をまとめた上で比較評価を行うこととする。

6.1.1 FIRST VIRTUAL

FIRSTVIRTUAL[18]は、専用の口座番号を使って、クレジットカード情報をやり取りする方式で、決済は一般に WWW (WorldWideWeb) を用いて行い、確認は電子メー

ルで行うため、専用のソフトウェアは必要としない。また、専用の口座番号を使用するので、特に暗号化はされていない。

利用者はまず、FIRST VIRTUAL にWWW で登録する。登録すると電子メールで電話番号が送られてくる。クレジットカード番号の登録は、この電話番号に電話し、プッシュボタンで登録する。しばらくすると専用の口座番号が電子メールで送られてくるので、決済にはこの口座番号を用いる。支払いをした場合、確認の電子メールが送られてくるので、身に覚えのない場合には"FRAUD"（不正使用）と返信すれば、口座が凍結され、被害を被ることはないという仕組みになっている。決済の際には必ずFIRST VIRTUAL のセンターが介在して、クレジットカード会社との間で決済情報をやり取りし、不正使用が通知された場合の口座の即時凍結などを可能としている。

既に実現されていて、実際の決済にも利用されている。専用のソフトウェアを必要としないためか、利用者数も多い。しかし、暗号化が行われないため、安全性は支払確認の通知や返信に利用される電子メールシステムの安全性に依存している。電子メールシステムについては、すでに多くの攻撃法が存在し、それに対する対処も進んでいるが、安全性が保証されているわけではない。そのため、FIRST VIRTUAL の安全性についても疑問の声が上がっている。

6.1.2 CyberCash

CyberCash[16] は、暗号化したクレジットカード情報をやり取りする方式で、専用のクライアントソフトウェアを使用する。決済や確認についても専用のクライアントソフトウェアを介して行われるので、利用者はクライアントソフトウェアを起動するだけで、実際に決済が行える。

利用者はまず、専用のクライアントソフトウェアをダウンロードし、インストゥールする。インストゥール時に、クレジットカード情報や個人情報の登録を求められ、登録した情報は暗号化されてCyberCash サーバに送られる。決済の際には必ずCyberCash サーバが介在して、支払人の取引確認と受取人への取引承認やクレジットカード会社との間での決済情報のやり取りが行われる。全てのやり取りは暗号化され、相手を特定できるために不正が行える余地は少ない。

既に実現されていて、実際の決済にも利用されている。実現されたのが早く、しかも暗号化により安全性を提供していたためか、利用者数も多い。しかし、最近ではNetscape

社の SSL（Secure Socket Layer）を利用することで、専用のソフトウェアを使用しなくてもクレジットカード情報を暗号化して取引ができるため、こちらの方を選択する利用者が増えているようである。

6.1.3 NetCheque

NetCheque[22] は、Kerberos[4]を利用して、電子的な小切手をやり取りする方式で、専用のソフトウェアは必要としない。決済は、電子メールで受取った小切手を譲渡することで行う。小切手には Kerberos による証明がなされている。

利用者はまず、取引サーバと Kerberos に登録する。登録の方法は、取引サーバの運営に依存する。利用者は決済の前に、小切手を書き（受取人や額面を指定する）、取引サーバを介して証明してもらう。決済は、この小切手を受取人に譲渡することで行う。小切手は取引サーバに預金するか現金引換えを要求することで現金化できる。この際に、取引サーバは Kerberos に小切手の検証を依頼する。二重使用などはこの際に発見することができる。取引サーバは、全ての取引を知るので、この方式は匿名性を提供できない。

既に実現されており、引き受け機関も存在した。引き受け機関との連絡が取れないため、現在も活動しているかどうかは確認できなかった。また、この方式は Kerberos システムに依存するためか、あまり普及していない。さらに、近年では Kerberos システムの安全性やスケーラビリティに対する疑問の声も上がっている。

6.1.4 ecash

ecash[17] は、ブラインド署名技術を利用して、発行局によって証明された通貨情報をやり取りする方式で、専用のクライアントソフトウェアを使用する。通貨情報は引き受ける金融機関によって兌換され、現金化することができる。

利用者はまず、専用のクライアントソフトウェアをダウンロードし、インストゥールする。インストゥール時に、引き受け金融機関上の口座情報や個人情報の登録を求められ、登録した情報は暗号化されて ecash の発行局に送られる。利用者は決済の前に発行局に通貨の発行を要求し、通貨を受取る。この際ブラインド署名が用いられ、利用者を特定することなく通貨に発行局の証明が与えられる。決済は、この通貨を受取人に譲渡することで行い、通貨を受取った受取人は直ちに通貨の信用照会を発行局に対し行う（オンライン

チェック)。この信用調査が必ず行われるために、二重使用などの不正を検出することができる。

既に実現されており、いくつかの銀行が実際に引き受けを行っている。ブラインド署名によって匿名性が提供されているなど、通貨としての体裁を持っており、金融機関がその引き受けを行うなど通貨としての信用も兼ね備えている。しかし、通貨情報が無断複製され使用された場合には、その匿名性ゆえに不正として検出することは難しいという欠点が指摘されている。また、匿名性があるがゆえに犯罪への利用やマネー・ロンダリングへの利用が懸念されるという問題もある。

6.1.5 MONDEX

MONDEX[20] は、耐タンパー性のある IC カードを利用して、通貨情報のやり取りを行う方式で、専用のハードウェアとそれに組み込まれた専用のソフトウェアを使用する。専用のハードウェアには、通貨情報を記録する IC カード「モンデックスカード」、電話を使って銀行口座からの入出金を行う「モンデックス電話機」、店頭での支払い、入金用端末「リテーラー端末」、カード間での通貨の譲渡を可能とする「ウォレット」、カードの残高を確認する「残高表示機」がある。決済は、「リテーラー端末」か「ウォレット」に「モンデックスカード」を挿入し、通貨情報を譲渡することで行う。

利用者はまず、発行局に登録し、「モンデックスカード」や必要となる専用の端末を受取る(もしくは購入する)。決済の前に利用者は、発行局に通貨を発行してもらい、「モンデックスカード」に保存する。この通貨を譲渡することによって、決済する。「モンデックスカード」は耐タンパー性を持っているため、その内容を複製することは非常に困難である。そのため、複製して不正使用することはできない。不正な通貨情報に関しては、専用端末によってオフラインチェックされる。

既に実現されており、商用実験がイギリスのスウィンドン市などで行われている。利用者の自宅に「モンデックス電話」があれば、24 時間いつでも銀行口座からの入出金が可能となる。国際的通用力は今のところないが、次世代の現金として期待されている。しかし、銀行と商店が結託することにより、匿名性が保証されなくなってしまうという問題が指摘されている。

6.1.6　供託電子マネー方式

供託電子マネー方式 [8][11][1] とは、鍵供託 [7] の機能を電子通貨システムに付加したもので、いくつかの方式が最近提案されている。この方式では、利用者が不正を行わない限りにおいては匿名性が保証されるが、不正や犯罪が発覚した場合、匿名性を無効化することができる。これによって、電子通貨の犯罪への利用やマネー・ロンダリングへの利用を抑止することができる。

いくつかの方式があるが、基本的には複数の受託者に対し、決済毎にもしくは何らかの条件にしたがって、通貨情報の匿名性を解除できるような鍵を寄託するという仕組みである。これによって、事後的に受託者すべてが情報を持ちよることで匿名性を解除することができ、不正者や裁判所などに証拠付きで不正の事実を示すことができる。

まだ提案段階の方式で、実現はされていないが、匿名性を保ちながら犯罪やマネー・ロンダリングへの使用を抑止することができるため、今後の電子通貨システムの主流となる可能性がある。しかし、受託者が結託することで匿名性を解除できるため、利用者がこの方式を受け入れるかどうかは受託者の社会的信用に依存すると考えられている。

6.2　比較評価の観点

比較評価の観点は、第 2 章の要件を念頭において考えることとする。

まず、兌換性・不換性であるが、これは、実現され実際に決済ができるかどうかで評価することとする。細かい評価を行うためには、実際にどの程度利用され、決済されているかを知らねばならないが、それはかなり困難であるためである。

次に、匿名性については、その決済システムにおいて匿名性が提供されているかどうかを見る。実際には、取引のどのような状況において、誰が誰についての情報を知り得ないのかを評価する必要があるが、そこまでの評価をしなくても大きな違いがあるので、このように評価することとする。

安全性については、取引が暗号化されているかどうかで判断する。細かい評価を行うためには、取引における通信の解析や使用している暗号系の強度への評価が必要となり、本研究のスコープから外れると思われるために、今回は行わない。

ユーザーインターフェイスについては、実際に利用者に比較利用してもらう必要があるため、今回は評価の対象とはしない。その代わりに、専用のハードウェアやソフトウェア

を必要とするかどうかといった観点を導入し、普及に関する評価を補強することとする。

スケーラビリティについては、主に通信負荷を考え、取引に常にサーバが介在する必要があるかどうかで評価することとする。実際には処理負荷なども考える必要があるが、実装やアルゴリズムに依存するので、今回は行わないこととする。

6.3 比較評価

前節の観点を箇条書きにまとめると以下のようになる。

実 現 実現されていて、決済が可能である
匿名性 取引の匿名性が提供されている
暗号化 取引が暗号化されている
ハード 専用のハードウエアを必要としない
ソフト 専用のソフトウェアを必要としない
サーバ 取引に常にサーバが介在する必要がない

この観点に基づいた星取表を表 6.1 に示す。星取表に従うと、本方式は実現すれば、匿名性に関して不利で、取引に常にサーバが介在する必要がないという優位点があるといえる。

方式	実 現	匿名性	暗号化	ハード	ソフト	サーバ
FIRST VIRTUAL	○	×	×	○	○	×
CyberCash	○	×	○	○	×	×
NetCheque	○	×	○	○	○	△[1]
ecash	○	○	○	○	×	×
MONDEX	△[2]	△[3]	○	×	×	○
供託電子マネー方式	×	△[4]	○	○	×	△[5]
本方式	△[6]	×	○	○	×	○

表 6. 1: 各電子決済方式の比較表

[1] サーバは階層化することができ、利用者は選択したサーバにのみアクセスする
[2] 現在一部の都市（イギリスのスウィンドン市）で商用実験が行なわれている
[3] 銀行と商店が結託することで匿名性が保持できなくなる
[4] サーバは介在しないが条件によって受託者を介在させる必要がある
[5] サーバと全ての受託者が結託することで取引内容の全てを知ることができる
[6] 実装はαテスト段階で、今後実験局の開設を行なう予定である

第 7 章

今後の課題

本方式の今後の課題としては、すでに第 5 章であげた問題点の克服が急務であると考えている。なぜなら、評価結果（第 6.3 節）に従うなら、これらの問題が克服されない限り、本方式の優位点だけでは有効な決済方式とは言えないと考えるからである。

また、サンプル実装は、第 4.4 節で述べたように、暗号化モジュール、通信モジュールともに安全性やスケーラビリティを考慮した設計に再設計し、実装する必要がある。また、ユーザーインターフェイスも暫定的なものに過ぎないので、再設計／実装を必要とする。つまり、現在行なっているサンプル実装はあくまでサンプルで、実際に運用する場合には、全く別の設計／実装を必要とする。

今後は、運用可能な実装を完了し、実験局を開設して、本方式の優位点の検証を行い、決済システムとしての完成を図るつもりである。決済システムとして完成した後には、引き受け機関を探し、実際に運用していきながら、インターネット上での通貨の効果について研究していきたいと考えている。

なお、第 8 章に示しすような本質的な問題がインターネット上での通貨には存在するので、これらの解決についても、今後の研究課題として考えていきたい。

第 8 章

議論

本研究を行うことによって提起された、インターネット上での通貨に関する重要な問題について議論する。

8.1 到達の確認

既に述べたとおり、インターネットは安全な通信路ではない。このような安全ではない通信路を使った取引においては、情報の到達の確認が大きな問題となる。特に商取引の場合、それが対面で行われる限り、代金を支払って商品を受け取っても、商品を受け取って代金を支払っても、その場で確認が可能であるが、安全ではない通信路を通して行われる場合、代金や商品の到達は、送った側では確認できない。ネットワーク層やトランスポート層のレベルでは確認することができるが、安全でない通信路においては盗難や改ざんやなりすましがありうるので、到達させたい人（本研究の提案方式でいえば、到達させたい情報を復号化できる秘密鍵を持った人）に到達したかどうかの確認にはならない。

そのため、受け取った側が受け取った事を示す確認情報を送る必要があるが、「受け取ったが受け取っていないように振る舞う」ことが可能である。これを防ぐためには、「受け取ったという事実が生じた時には、必ず確認情報が送り手に届く」必要がある。これを実現するためには、受け取った側の制御によらず確認情報を送ることができる必要がある。これはかなり困難であると考えられる。

そこで、本質的ではない解決であるが、提案方式の裏書譲渡のような方法（第 3.3. 2節）が考えられる。この方法では、受取人から何ら「裏書譲渡」を行っていた証拠となる情報

を受け取らずに、銀行券本体を送った場合、確認情報が得られなかったときに、支払人は相手の署名が施されたセッション ID しか持たないため、「裏書譲渡」を行っていたことを証明することはできないという問題に対し、あらかじめ確認情報（領収書）をもらい、それから銀行券本体を送信するようにすることで解決を試みる。ただし、確認情報（領収書）には、何らかの不正に利用できるような価値はないとする。そして、補助的に銀行券本体到達後、受取人は受取確認情報を送る。支払人は、受取人の請求にしたがって、保存してある銀行券本体を再送する。この方法なら、「裏書譲渡」を行っていた証拠として「領収書」があるので、それを根拠として商品を請求することができる。その際に、銀行券本体が到達していないことが判明した場合、支払人は保存してある銀行券本体を再送すればよい。再送は何ら利益／不利益を伴わないので、支払人／受取人の不正の有無に関わらず行うことができる。

しかし、この方法では銀行券本体や受取確認情報が通信経路上で盗難されるなどによって失われた場合、受取確認情報は支払人に届かないので、結局銀行券本体が到達したかどうかは知りようがない。つまり、最終的な「代金到達の確認」は何らか別の方法を必要とするために、本質的な解決とはならない。

本質的な解決は、安全な通信路を用いることであるが、インターネット上での商取引の決済に別の特殊な通信路を使うことは現実的とはいえない。そこで、インターネット自身が安全な通信路となる必要がある。これについては、インターネットの次世代プロトコルである IPv6 では、安全性について配慮されていると思われるので、その実現に期待したい。

8.2 最終的決済の可能性

現在のインターネット上での通貨は、決済手段であり、多くは決済情報のやり取りに過ぎなず、最終的決済は現実の決済機関（銀行やカード会社）を通して行われる。つまり、兌換通貨である。兌換通貨のおおきな欠点は二つある。一つは通貨の能力が最終的決済で束縛されてしまう事。もう一つは、最終的決済を行う際にコストがかかる事である。

インターネット上での通貨に求められるのは、24 時間決済や国際決済といった要求に応えることと、決済コストを軽減することである。しかし、最終的決済が行えない既存の方式では、これらの要求に完全に応えることはできない。なぜなら、兌換は 24 時間可能とは限らず、また国際決済は別途行う必要があるかもしれない。その上、最終的決済の際

には何らかの手数料を必要とし、引き受け機関は兌換のためのシステムの設備投資を必要とするため、決済コストが軽減するとは限らないからである。

そこで、本方式では最終的決済を行うことを目指し、インターネット上で兌換されることなく流通しつづける可能性があるような、転々譲渡機能を持った電子銀行券方式を提案した。しかし、本方式の銀行券が最終的決済として利用されるためにはいくつかの条件があると考えられる。これは、転々譲渡機能によって最終的決済を目指す限り、避けては通れない問題となると考えている。

通貨が兌換されること無しに最終的決済が可能であるということは、その通貨は不換通貨（第 2.1.1 節）であるということである。ある通貨が不換通貨となるためには、通貨の発行を社会的に信用の高い機関（一般的には、安定した政府による裏付けを伴う中央銀行）が行い、かつその通貨が市場で流通し、実際の市場での決済を行える必要がある。

本方式を安定した政府による裏付けを伴った中央銀行が引き受けた場合を考える。本方式は、通貨を流通させるために転々譲渡機能を持っている。しかし、転々譲渡機能を付加することは、通貨の流通を可能とするということではあるが、これは流通が行われることを示すわけではない。そのため、実際に流通が行われるかどうかは別の要素に依存すると考えられる。本方式の通貨の信用が充分に高い場合、それは市場での通用力である。通用力とは、市場においてどのような場面でも決済に使用できる力のことである。通用力を決定するのは、通貨の信用と市場での流通量である。つまり、通貨の信用が充分に高い場合、通用力は通貨の流通量に従う。これは、前述の流通が通用力に依存するということとの間で、「鶏が先か卵が先か」問題を生ずることを意味している。

また、社会的に信用の高い機関は充分に通用力を持ち、流通している通貨以外について引き受けを行うことはないと考えられる。このようなことを考えると、最終的決済が可能となるための敷居は非常に高いといわざるを得ない。さらに大きな問題として、インターネットのような現実の経済や国家の枠組みを超えたものの上で流通する通貨の引き受けは、現実の経済や国家の枠組みを超えた経済力を必要とする可能性が高い。以上のことから、本研究では転々譲渡機能を付加することによって、最終的決済が可能な通貨を目指し、24 時間決済や国際決済、決済コストの軽減といった要求に応えることを目指したが、残念ながらその実現はかなり困難であるといわざるを得ないだろう。

第 9 章

インターネット上での通貨に関する展望

インターネット上での兌換通貨 [17][19] は、近年の内にいくつかの標準を得て、一般的な決済に使用されるようになっていくだろう。これらが現実の電子現金と結び付いていくことで現在の経済や国家の枠組に対してどのようなインパクトを与えていくかについては、今後注目すべきことである。

インターネット上での通貨に関しては、大きな問題が存在することは前述のとおりである。しかし、「到達の確認」（第 8.1 節）については、インターネット自身が安全性や到達性を保証できるように変化することで解決が可能であると考えている。また、「最終的決済の可能性」（ 第 8. 2節）については、近年、超国家的に行われている電子現金のプロジェクトと結びつくこと（すでに MONDEX[20] などは、インターネット上で利用可能なシステムについて検討している）によって解決への糸口が見えるのではないかと考えている。

もしこれらの問題が解決されたならば、インターネット上での通貨は現実の経済や国家の枠組みを超えて活動し、インターネットはそのコミュニティに独自の経済を手に入れることとなるだろう。21 世紀に向かって現実社会の既存の枠組みが崩壊もしくは再編成しながら昏迷する中、インターネットは現実の社会に対し、新たな枠組みのモデルを示すことができるかもしれない。

本研究が「インターネット上での通貨」を成り立たせる上で、何らかの力となれば幸いである。

謝辞

研究にあたって、指導教官である篠田 陽一助教授（北陸先端科学技術大学院大学情報科学研究科）には、様々な助言、指導を頂いた。研究における様々な躓きから脱することができたのみならず、今後の研究生活にとって大きな糧となった。金沢大学経済学部の佐々木 雅幸教授には、「通貨と価値」に関して経済学的見解を助言して頂き、本研究に経済学的視点を盛り込むにあたって、大きな助けとなった。所属研究室の博士後期課程 3 年生であった佐藤 円氏には、文献を国外において検索して頂いた。研究の初期においての手がかりとなった。また、所属研究室のその他の方々からも様々な意見を頂いた。記して感謝の意を示す。

最後に、私の研究とその資金と生活を支えてくれた妻への感謝をもって、論文を締めくくりたい。

参考文献

[1] J. Camenish, J. M.Piveteau, and M Stadler. **An Efficient Fair Payment System** In *Proceedings of 3rd ACM Conference on Computer Communications Security* 1996.

[2] D. Chaum, B. Boer, E. Heyst, S. Mjolsnes, and A. Steenbeek. **Efficient off-line checks**. In *Proceedings of Eurocrypt '89*, 1989.

[3] D. Chaum, A Fiat, and N. Naor. **Untraceable electronic cash** In *Proceedings of Crypto '88*, 1988.

[4] B. Clifford Neuman and Theodore Ts'o. **Kerberos: An authentication service for computer networks** *IEEE Communications, 32(9)*, September 1994.

[5] W. Diffie and M.E. Hellman. **New Directions in Cryptography** *IEEE Transactions on Information Theory, v.IT-22, n. 6, pp. 644-654*, November 1976.

[6] S. Even, O. Goldreich, and Y. Yacobi. **Electroni Wallet** In *Proceedings of Crypto '83*, 1983.

[7] 満保 雅浩, 岡本 栄司. **暗号最新事情.2 ネットワーク暗号クリッパーのインパクト－鍵供託方式に関する技術的な論争－**. *bit, vol.28, NO.2, pp.53-61*, 1996.

[8] 満保 雅浩, 岡本 栄司. **暗号最新事情.9 供託電子マネー方式**. *bit, vol.28, NO.9, pp.101-109*, 1996.

[9] National Bureau of Standards, U.S Department of Commerce. **Data Encryption Standard** *Federal Information Processing Standards Publication 46 (FIPS PUB 46), Washington, DC*, 1977.

[10] T. Okamoto and K. Ohta. **Universal electronic cash**. In *Proceedings of Crypto '91*, 1991.

[11] E. Fujiaki, and T. Okamoto. **Practical Escrow Cash Systems**. *ISEC 95-46*, March 1996.

[12] R. L. Rivest, A. Shamir, L. M. Adleman. **A Method for Obtaining Digital Signatures and Public-Key Cryptosystems**. *Communications of the ACM, Vol.21, pp. 120-126*, February 1978.

[13] D. Atkins, W. Stallings, and P. Zimmermann. **PGP Message Exchange Formats**. *RFC1991*, 1996.

[14] R. Baldwin and R. L. Rivest. **The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms**. *RFC2040*, October 1996.

[15] RSA Laboratories. **Public-Key Cryptography Standards, #7: Cryptographic Message Syntax Standard**. *An RSA Laboratories Technical Note Version 1.5*, November 1993.

[16] CyberCash Home Page. ⟨URL: http://www.cybercash.com/⟩.

[17] DigiCash home page. ⟨URL: http://www.digicash.com/⟩.

[18] FV HOMEPAGE. ⟨URL: http://www.fv.com/⟩.

[19] FSTC Project. ⟨URL: http://www.fstc.org/projects/projects.html⟩.

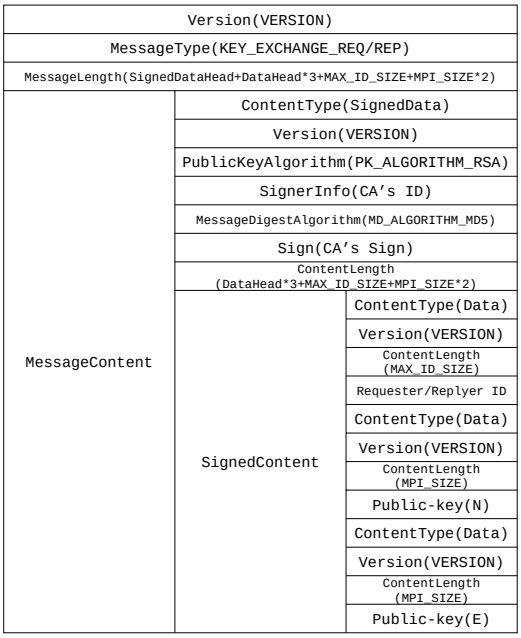
[20] Mondex Home Page. ⟨URL: http://www.mondex.com/index.html⟩

[21] The USC NetCash anonymous network payment researh prototype. ⟨URL: http://gost.isi.edu/info/netcash/

[22] The NetCheque(SM) network payment system ⟨URL: http://nii-server.isi.edu/info/NetCheque/

付録 A

各メッセージの構造



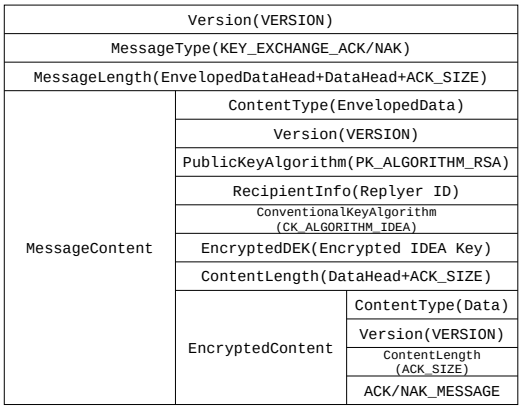


図 A.1: 鍵交換 REQ/REP/ ACK/ NAKメッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ISSUE_REQ)</td></tr>
<tr><td colspan="4">MessageLength(EnvelopedDataHead+SignedDataHead+DataHead+4)</td></tr>
<tr><td rowspan="14">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Issuer’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength(SignedDataHead+DataHead+4)</td></tr>
<tr><td rowspan="9">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(User’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(User’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead+4)</td></tr>
<tr><td rowspan="4">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength(4)</td></tr>
<tr><td>(Amount)</td></tr>
</table>

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ISSUE_CHA/RES)</td></tr>
<tr><td colspan="4">MessageLength(EnvelopedDataHead+SignedDataHead+DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="14">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(User’s/Issuer’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength(SignedDataHead+DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="9">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Issuer’s/User’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Issuer’s/User’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="4">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength (IDEAKEYSIZE)</td></tr>
<tr><td>(ID)</td></tr>
</table>

図 A.2: 振出 REQ/CHA/ RESメッセージの構造

<table>
<tr><td colspan="6">Version(VERSION)</td></tr>
<tr><td colspan="6">MessageType(ISSUE_REP)</td></tr>
<tr><td colspan="6">MessageLength(EnvelopedDataHead+EncryptedDataHead+SignedDataHead*5+DataHead*6+4*4+IDEAKEYSIZE*2)</td></tr>
<tr><td rowspan="25">MessageContent</td><td colspan="5">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="5">Version(VERSION)</td></tr>
<tr><td colspan="5">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="5">RecipientInfo(User’s ID)</td></tr>
<tr><td colspan="5">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="5">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="5">ContentLength(EncryptedDataHead+SignedDataHead*5+DataHead*6+4*4+IDEAKEYSIZE*2)</td></tr>
<tr><td rowspan="24">EncryptedContent</td><td colspan="4">ContentType(EncryptedData)</td></tr>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="4">RecipientInfo(Issuer’s ID)</td></tr>
<tr><td colspan="4">ContentLength(SignedDataHead*2+DataHead*3+4*2+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="23">EncryptedContent</td><td colspan="3">ContentType(SignedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">SignerInfo(Issuer’s ID)</td></tr>
<tr><td colspan="3">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="3">Sign(Issuer’s Sign)</td></tr>
<tr><td colspan="3">ContentLength(SignedDataHead+DataHead*3+4*2+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="22">SignedContent</td><td colspan="2">ContentType(Data)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">ContentLength(4)</td></tr>
<tr><td colspan="2">(Number)</td></tr>
<tr><td colspan="2">ContentType(Data)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">ContentLength(4)</td></tr>
<tr><td colspan="2">(Amount)</td></tr>
<tr><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(User’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(User’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="4">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength (IDEAKEYSIZE)</td></tr>
<tr><td>(ID)</td></tr>
<tr><td colspan="4">FaceValueInfo</td></tr>
</table>

図 A.3: 振出 REP メッセージの構造

<table>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">MessageType(ISSUE_ACK)</td></tr>
<tr><td colspan="3">MessageLength(EnvelopedDataHead+SignedDataHead*3+DataHead*3+4*2+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="8">MessageContent</td><td colspan="2">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">RecipientInfo(Issuer’s ID)</td></tr>
<tr><td colspan="2">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="2">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="2">ContentLength(SignedDataHead*3+DataHead*3+4*2+IDEAKEYSIZE)</td></tr>
<tr><td>EncryptedContent</td><td>FaceValueInfo</td></tr>
</table>

図 A.4: 振出 ACK メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ENDORSE_CHA)</td></tr>
<tr><td colspan="4">MessageLength(EnvelopedDataHead+SignedDataHead+DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="13">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Endorser’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength(SignedDataHead+DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="11">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Endorsee’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Endorsee’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead+IDEAKEYSIZE)</td></tr>
<tr><td rowspan="4">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength
(IDEAKEYSIZE)</td></tr>
<tr><td>(Endorsee’s RAND)</td></tr>
</table>

図 A.5: 裏書譲渡 CHA メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ENDORSE_RES)</td></tr>
<tr><td colspan="4">MessageLength
(EnvelopedDataHead+SignedDataHead*4+DataHead*4+IDEAKEYSIZE*3+4)</td></tr>
<tr><td rowspan="19">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Endorsee’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength
(SignedDataHead*4+DataHead*4+IDEAKEYSIZE*3+4)</td></tr>
<tr><td rowspan="18">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Endorser’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Endorser’s Sign)</td></tr>
<tr><td colspan="2">ContentLength
(SignedDataHead*2+DataHead*3+IDEAKEYSIZE*3)</td></tr>
<tr><td>SignedContent</td><td>SessionID</td></tr>
<tr><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Issuer’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Issuer’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead+4)</td></tr>
<tr><td rowspan="4">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength(4)</td></tr>
<tr><td>(Amount)</td></tr>
</table>

図 A.6: 裏書譲渡 RES メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ENDORSE_REQ)</td></tr>
<tr><td colspan="4">MessageLength(EnvelopedDataHead+SignedDataHead*3+DataHead*3+IDEAKEYSIZE*3)</td></tr>
<tr><td rowspan="11">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Endorser’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength
(SignedDataHead*3+DataHead*3+IDEAKEYSIZE*3)</td></tr>
<tr><td rowspan="8">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Endorsee’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Endorsee’s Sign)</td></tr>
<tr><td colspan="2">ContentLength
(SignedDataHead*2+DataHead*3+IDEAKEYSIZE*3)</td></tr>
<tr><td>SignedContent</td><td>SessionID</td></tr>
</table>

図 A.7: 裏書譲渡 REQ メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ENDORSE_REP)</td></tr>
<tr><td colspan="4">MessageLength(EnvelopedDataHead+SignedDataHead+DataHead*2+IDEAKEYSIZE+4)</td></tr>
<tr><td rowspan="18">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Endorsee’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength(SignedDataHead+DataHead*2+IDEAKEYSIZE+4)</td></tr>
<tr><td rowspan="17">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Issuer’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Issuer’s Sign)</td></tr>
<tr><td colspan="2">ContentLength
(DataHead*2+IDEAKEYSIZE+4)</td></tr>
<tr><td rowspan="9">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength
(IDEAKEYSIZE)</td></tr>
<tr><td>(ID)</td></tr>
<tr><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength(4)</td></tr>
<tr><td>(Number)</td></tr>
</table>

図 A.8: 裏書譲渡 REP メッセージの構造

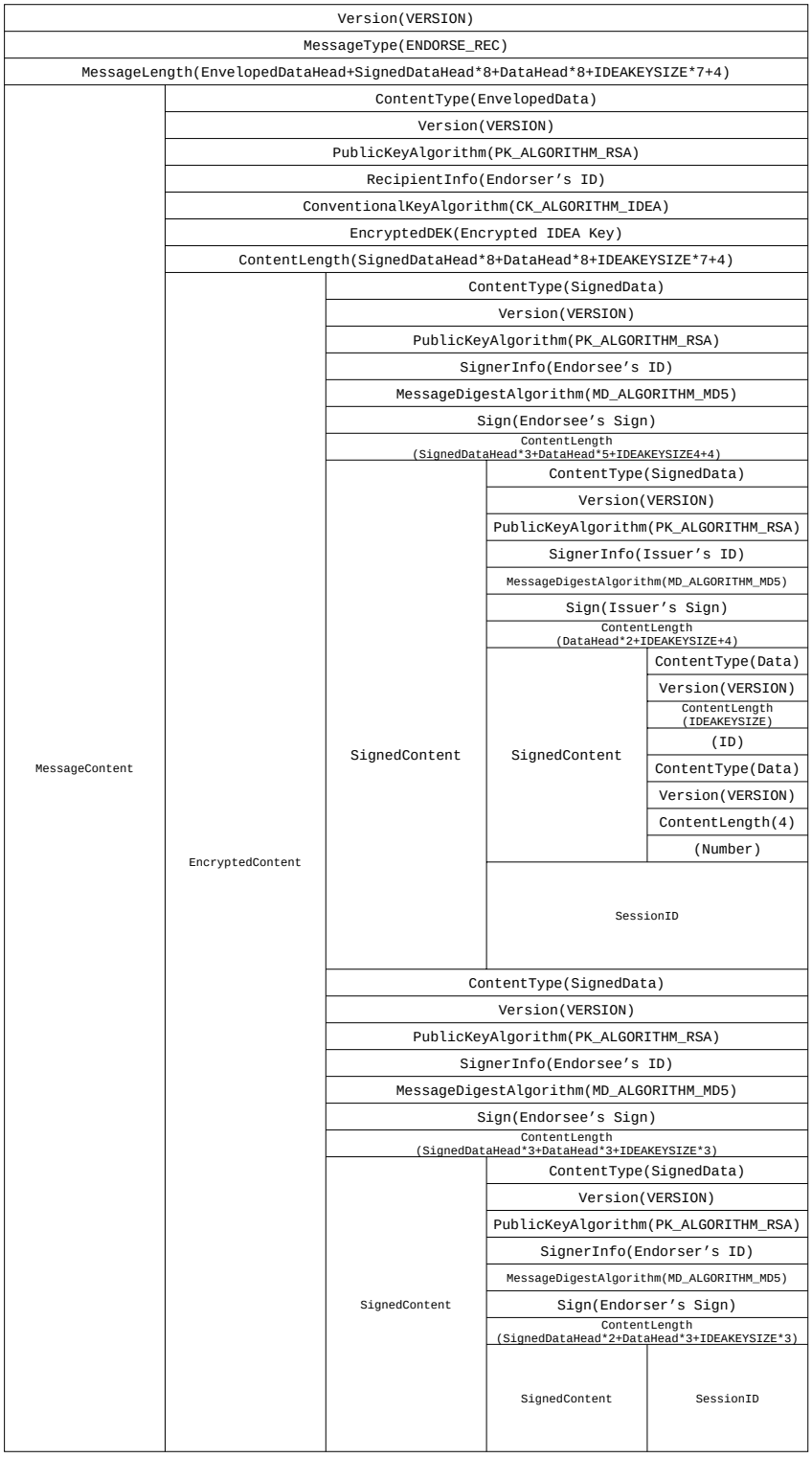


図 A.9: 裏書譲渡 REC メッセージの構造

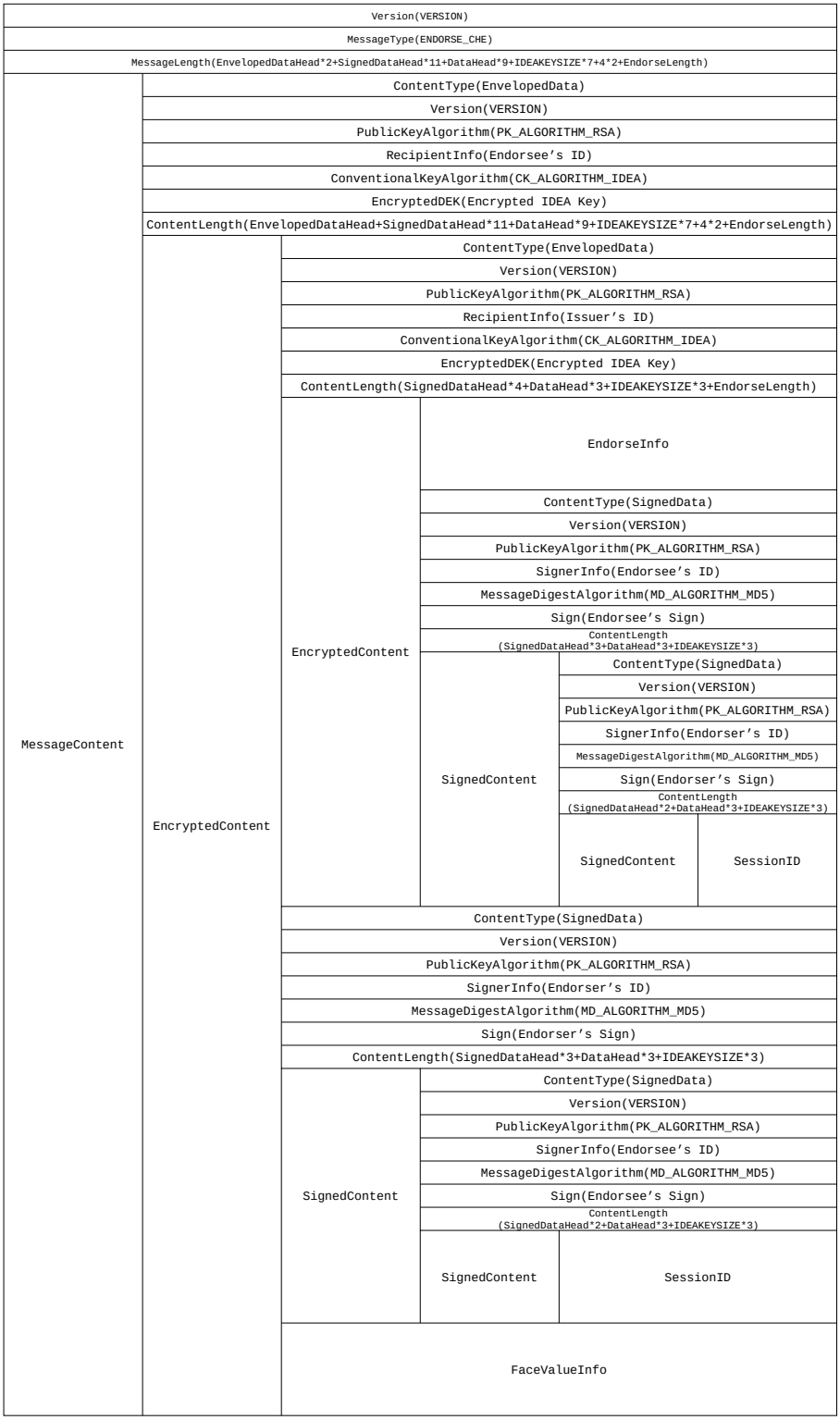


図 A.10: 裏書譲渡 CHE メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ENDORSE_ACK)</td></tr>
<tr><td colspan="4">MessageLength
(EnvelopedDataHead+SignedDataHead*6+DataHead*6+IDEAKEYSIZE*4+4*2)</td></tr>
<tr><td rowspan="14">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Endorser’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength
(SignedDataHead*6+DataHead*6+IDEAKEYSIZE*4+4*2)</td></tr>
<tr><td rowspan="8">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Endorsee’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Endorsee’s Sign)</td></tr>
<tr><td colspan="2">ContentLength
(SignedDataHead*5+DataHead*6+IDEAKEYSIZE*4+4*2)</td></tr>
<tr><td rowspan="2">SignedContent</td><td>FaceValueInfo</td></tr>
<tr><td>SessionID</td></tr>
</table>

図 A.11: 裏書譲渡 ACK メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(ISSUE/ENDORSE_NAK)</td></tr>
<tr><td colspan="4">MessageLength(EnvelopedDataHead+SignedDataHead+DataHead+ACK_SIZE)</td></tr>
<tr><td rowspan="14">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Issuer’s/User’s/Endorser’s/Endorsee’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength(SignedDataHead+DataHead+ACK_SIZE)</td></tr>
<tr><td rowspan="10">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo
(User’s/Bank’s/Endorsee’s/Endorser’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign
(User’s/Bank’s/Endorsee’s/Endorser’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead+ACK_SIZE)</td></tr>
<tr><td rowspan="4">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength
(ACK_SIZE)</td></tr>
<tr><td>NAK_MESSAGE</td></tr>
</table>

図 A.12: 振出 NAK、裏書譲渡 NAK メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(CHECK_REQ)</td></tr>
<tr><td colspan="4">MessageLength
(EnvelopedDataHead+SignedDataHead+DataHead*(N+1)+4*(N+1)+BankNoteLength)</td></tr>
<tr><td rowspan="15">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(Issuer’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength
(SignedDataHead+DataHead*(N+1)+4*(N+1)+BankNoteLength)</td></tr>
<tr><td rowspan="13">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(User’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(User’s Sign)</td></tr>
<tr><td colspan="2">ContentLength(DataHead*(N+1)+4*(N+1))</td></tr>
<tr><td rowspan="5">SignedContent</td><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength(4)</td></tr>
<tr><td>(Amount)</td></tr>
<tr><td>Div1...DivN
(IF N=0, Not Exist)</td></tr>
<tr><td colspan="2">BankNoteInfo</td></tr>
</table>

図 A.13: 検証／引換 REQ メッセージの構造

<table>
<tr><td colspan="4">Version(VERSION)</td></tr>
<tr><td colspan="4">MessageType(CHECK_ACK/NAK)</td></tr>
<tr><td colspan="4">MessageLength
(EnvelopedDataHead+SignedDataHead+DataHead+ACK_SIZE+FaceValueLength)</td></tr>
<tr><td rowspan="15">MessageContent</td><td colspan="3">ContentType(EnvelopedData)</td></tr>
<tr><td colspan="3">Version(VERSION)</td></tr>
<tr><td colspan="3">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="3">RecipientInfo(User’s ID)</td></tr>
<tr><td colspan="3">ConventionalKeyAlgorithm(CK_ALGORITHM_IDEA)</td></tr>
<tr><td colspan="3">EncryptedDEK(Encrypted IDEA Key)</td></tr>
<tr><td colspan="3">ContentLength
(SignedDataHead+DataHead+ACK_SIZE+FaceValueLength)</td></tr>
<tr><td rowspan="9">EncryptedContent</td><td colspan="2">ContentType(SignedData)</td></tr>
<tr><td colspan="2">Version(VERSION)</td></tr>
<tr><td colspan="2">PublicKeyAlgorithm(PK_ALGORITHM_RSA)</td></tr>
<tr><td colspan="2">SignerInfo(Issuer’s ID)</td></tr>
<tr><td colspan="2">MessageDigestAlgorithm(MD_ALGORITHM_MD5)</td></tr>
<tr><td colspan="2">Sign(Issuer’s Sign)</td></tr>
<tr><td colspan="2">ContentLength
(DataHead+ACK_SIZE+FaceValueLength)</td></tr>
<tr><td rowspan="5">SignedContent</td><td>FaceValueInfo</td></tr>
<tr><td>ContentType(Data)</td></tr>
<tr><td>Version(VERSION)</td></tr>
<tr><td>ContentLength
(ACK_SIZE)</td></tr>
<tr><td>ACK/NAK_MESSAGE</td></tr>
</table>

図 A.14: 検証／引換 ACK/NAK メッセージの構造