

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4326189号  
(P4326189)

(45) 発行日 平成21年9月2日(2009.9.2)

(24) 登録日 平成21年6月19日(2009.6.19)

(51) Int.Cl.		F I		
<b>G06K 19/07</b>	<b>(2006.01)</b>	G06K 19/00		N
<b>G06K 19/073</b>	<b>(2006.01)</b>	G06K 19/00		P

請求項の数 8 (全 105 頁)

(21) 出願番号	特願2002-169315 (P2002-169315)	(73) 特許権者	592146793
(22) 出願日	平成14年6月10日(2002.6.10)		坂村 健
(65) 公開番号	特開2004-13748 (P2004-13748A)		東京都品川区大崎4-9-2
(43) 公開日	平成16年1月15日(2004.1.15)	(73) 特許権者	502180015
審査請求日	平成17年5月18日(2005.5.18)		越塚 登
前置審査			東京都武蔵野市西久保2-27-20
		(73) 特許権者	392026693
			株式会社エヌ・ティ・ティ・ドコモ
			東京都千代田区永田町二丁目11番1号
		(74) 代理人	100083806
			弁理士 三好 秀和
		(72) 発明者	坂村 健
			東京都品川区大崎4-9-2
		(72) 発明者	越塚 登
			東京都武蔵野市西久保2-27-20
			最終頁に続く

(54) 【発明の名称】 自律型 IC カード及び通信システム

(57) 【特許請求の範囲】

【請求項 1】

他の自律型 IC カードを含む通信相手装置とネットワークを介して接続された IC カード端末を経由して、前記他の自律型 IC カードと価値情報を送受信する自律型 IC カードであって、

前記 IC カード端末と物理層を介して接続されるインターフェースと、

前記他の自律型 IC カードと前記 IC カード端末を介して相互に認証を行う相互認証を行い、前記相互認証が成功した場合に、前記他の自律型 IC カードとセッションを構築し、前記セッションを介し、ネットワーク層のプロトコルを用いて前記他の自律型 IC カードと暗号化通信を行う IC チップと

を備え、

前記 IC チップ及び前記他の自律型 IC カードのそれぞれは、割り当てられた固有の識別子を有し、

前記固有の識別子は、前記ネットワーク層のプロトコルを用いた暗号化通信においてネットワーク層アドレスを構成し、

前記 IC チップは、前記価値情報を前記他の自律型 IC カードに送信する際に、前記他の自律型 IC カードのみが復号可能な暗号化を前記価値情報に施すことを特徴とする自律型 IC カード。

【請求項 2】

前記 IC チップは、前記他の自律型 IC カードに対して前記セッションを構築するセッ

10

20

ション管理命令群と、前記他の自律型 IC カードに対して、前記価値情報の送受信に関する処理をトランザクションによって行うためのトランザクションセッションを構築するトランザクション管理命令群と、を有することを特徴とする請求項 1 に記載の自律型 IC カード。

【請求項 3】

前記 IC チップは、前記他の自律型 IC カードと相互に認証処理を行うと共に、前記他の自律型 IC カードとの通信に係る情報に対して暗号化/復号化を施す暗号処理部を有することを特徴とする請求項 1 に記載の自律型 IC カード。

【請求項 4】

前記識別子に基づき、前記他の自律型 IC カードを識別し、相互の認証処理を行うことを特徴とする請求項 3 に記載の自律型 IC カード。

10

【請求項 5】

前記暗号処理部は、前記他の自律型 IC カードの種別に応じて、複数の認証処理及び複数の暗号処理の中から適した認証処理及び暗号処理を選択してそれらの処理を行うことを特徴とする請求項 3 に記載の自律型 IC カード。

【請求項 6】

前記 IC チップは、前記価値情報が格納される記憶部を有し、前記他の自律型 IC カードと前記価値情報に係る通信を行うことを特徴とする請求項 1 乃至 5 のいずれか 1 項に記載の自律型 IC カード。

【請求項 7】

前記インターフェースと、前記 IC カード端末とは、非接触で接続されることを特徴とする請求項 1 乃至 6 のいずれか 1 項に記載の自律型 IC カード。

20

【請求項 8】

自律型 IC カードと、  
他の自律型 IC カードを含む通信相手装置とネットワークを介して接続される IC カード端末と

を備える通信システムであって、

前記自律型 IC カードは、  
前記 IC カード端末と物理層を介して接続されるインターフェースと、  
IC チップとを備え、

30

前記 IC チップは、前記 IC カード端末を介して前記他の自律型 IC カードと相互に認証を行う相互認証を行い、前記相互認証が成功した場合に、前記他の自律型 IC カードとセッションを構築し、前記セッションを介し、ネットワーク層のプロトコルを用いて前記他の自律型 IC カードと暗号化通信を行い、

前記 IC カード端末は、前記 IC チップ及び前記他の自律型 IC カードが実行する前記暗号化通信を中継し、

前記 IC チップ及び前記他の自律型 IC カードのそれぞれは、割り当てられた固有の識別子を有し、

前記固有の識別子は、前記ネットワーク層のプロトコルを用いた暗号化通信においてネットワーク層アドレスを構成し、

40

前記 IC チップは、前記価値情報を前記他の自律型 IC カードに送信する際に、前記他の自律型 IC カードのみが復号可能な暗号化を前記価値情報に施すことを特徴とする通信システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、IC カードに関し、特に、IC カード端末に直接又はネットワークを介して接続される他の装置と、直接かつ自律的に通信を行う自律型 IC カードに関する。

【0002】

【従来の技術】

50

図4は、従来におけるICカードとICカード端末との信号のやり取りを説明するための図である。

ここで、ICカード端末は、ICカードリーダー/ライター200とICカード端末本体300とで構成される。尚、ここではICカード100とICカードリーダー/ライター200とは模式的に離れて示されているが、このように非接触型であってもよいし、勿論、ICカード100がICカードリーダー/ライター200に典型的には挿入される接触型であってもよい。

【0003】

このようなICカード100を利用した処理においては、ICカード端末本体300がマスター装置であり、ICカード100がスレーブ装置となっている。すなわち、ICカード100を利用した処理を行いたい場合、先ずICカード端末本体300に備えられた入力端末（図示せず）に操作者は指示入力する。それに応じて、マスター装置たるICカード端末本体300は、ICカードリーダー/ライター200を介して命令をICカード100に送る。その命令を受け取ったスレーブ装置たるICカード100はその命令に応じた処理を行い、その実行結果をICカードリーダー/ライター200を介してICカード端末本体300に返す。

10

【0004】

図5は、従来において、2枚のICカードが互いに通信を行う場合の信号のやり取りを示す図である。

同図において、図4に示した構成に加えて、ICカードリーダー/ライター200bがICカード端末本体300に接続されており、そのICカードリーダー/ライター200bにICカード100bが接触式又は非接触式で接続される形態になっている。

20

【0005】

そこで、図4の場合と同様、先ずICカード端末本体300に備えられた入力端末（図示せず）に操作者はカード間通信に係る指示入力する。それに応じて、ICカード端末本体300は、ICカードリーダー/ライター200aを介して命令をICカード100aに送る。その命令を受け取ったICカード100aはその命令に応じた処理を行い、その結果である返答Aを出力する。ICカード100aから出力された返答Aは、ICカードリーダー/ライター200aを介してICカード端末本体300に入力される。ICカード端末本体300は、その返答Aたる情報に対しての中継動作を行う。ICカード端末本体300から出力された返答Aは、ICカードリーダー/ライター200bを介してICカード100bに入力される。ICカード100bは、入力された返答Aに応じた処理を行い、その結果として返答Bを出力する。ICカード100bから出力された返答Bは、ICカードリーダー/ライター200bを介してICカード端末本体300に入力される。ICカード端末本体300は、その返答Bたる情報に対しての中継動作を行う。ICカード端末本体300から出力された返答Bは、ICカードリーダー/ライター200aを介してICカード100aに入力される。

30

【0006】

以上のようなICカード100aとICカード100bとの間のやりとりが更に続いても、基本的には動作は同じである。すなわち、ICカード端末本体300は、ICカード100a ICカード100bとの間でやりとりされる情報に対する中継動作を行うという役目を必ず担っている。

40

【0007】

【発明が解決しようとする課題】

ところで、従来におけるICカード間通信においては、上述のようにICカード端末本体が必ずICカード間通信に係る情報の中継動作を行っている。従って、従来においては、ICカード端末本体に代表される介在装置が、ICカード間の通信情報を改竄したり盗聴したりすることが可能であった。故に、従来においては、ICカード間の正確な情報の通信が保証されないという課題があった。

【0008】

50

本発明は、上述のような事情を鑑みて為されたものであり、本発明の目的は、ＩＣカードがＩＣカード端末に直接又はネットワークを介して接続される他の通信装置と通信を行うに際し、ＩＣカードと他の通信装置との間に介在するＩＣカード端末等の装置を見かけ上介在させることなく、他の通信装置と直接通信を行うことにより、正確な情報の安全な通信を保證することができる自律型ＩＣカードを提供することにある。

【 0 0 0 9 】

【課題を解決するための手段】

上記目的を達成するため、請求項１の自律型ＩＣカードは、ＩＣカード端末と物理層を介して接続される論理ホストインターフェースと、前記ＩＣカード端末を物理的に介して接続される通信装置と通信を行うための論理外部通信インターフェースと、前記論理外部通信インターフェースを介し、前記通信装置の接続を認識し、前記通信装置と直接的に通信を行うように構成されたＩＣチップと、を備えることを要旨とする。

10

【 0 0 1 0 】

本発明によれば、本発明の自律型ＩＣカードは、ＩＣカード端末を介して接続される通信装置を自ら認識し、その通信装置と直接的に通信を行うことができる。

【 0 0 1 1 】

また、請求項２の発明は、請求項１の自律型ＩＣカードにおいて、前記ＩＣチップは、前記論理外部通信インターフェースを介して前記通信装置と直接的に通信を行うためのソフトウェアモジュールたる通信制御部を有することを要旨とする。

【 0 0 1 2 】

本発明によれば、通信制御部は、接続されているＩＣカード端末を意識しないソフトウェアモジュールによりその処理を行うことができる。

20

【 0 0 1 3 】

また、請求項３の発明は、請求項２の自律型ＩＣカードにおいて、前記通信制御部は、前記通信装置に対してセッション通信路を構築するセッション管理命令群と、前記通信装置に対してトランザクションセッションを構築するトランザクション管理命令群と、を有することを要旨とする。

【 0 0 1 4 】

また、請求項４の発明は、請求項１の自律型ＩＣカードにおいて、前記ＩＣチップは、前記通信装置と相互に認証処理を行うと共に、前記通信装置との通信に係る情報に対して暗号化／復号化を施す暗号処理部を有することを要旨とする。

30

【 0 0 1 5 】

本発明によれば、暗号処理部は、通信装置と相互に認証処理を行うと共に、通信装置との通信に係る情報に対して暗号化／復号化を施す。

【 0 0 1 6 】

また、請求項５の発明は、請求項４の自律型ＩＣカードにおいて、前記ＩＣチップは、割り当てられた固有の識別子を有し、その識別子に基づき、前記通信装置を識別し、相互の認証処理を行うことを要旨とする。

【 0 0 1 7 】

また、請求項６の発明は、請求項４の自律型ＩＣカードにおいて、前記暗号処理部は、認識された前記通信装置の種別に応じて、複数の認証処理及び複数の暗号処理の中から適した認証処理及び暗号処理を選択してそれらの処理を行うことを要旨とする。

40

【 0 0 1 8 】

また、請求項７の発明は、請求項１乃至６の自律型ＩＣカードにおいて、前記ＩＣチップは、価値情報が格納される記憶部を有し、前記他の通信装置とその価値情報に係る通信を行うことを要旨とする。

【 0 0 1 9 】

また、請求項８の発明は、請求項１乃至７の自律型ＩＣカードにおいて、前記ＩＣカード端末と前記他の通信装置は、ネットワークを介して接続されていることを要旨とする。

【 0 0 2 0 】

50

また、請求項 9 の発明は、請求項 1 乃至 8 の自律型 IC カードにおいて、前記論理ホストインターフェース及び前記論理外部通信インターフェースと、前記 IC カード端末とは、非接触で接続されることを要旨とする。

【 0 0 2 1 】

【発明の実施の形態】

以下、図面に基づいて、本発明における自律型 IC カードの実施形態について詳細に説明する。

【 0 0 2 2 】

図 1 は、本発明における自律型 IC カードの一実施形態の論理的な構成を示す図である。

【 0 0 2 3 】

図 1 において、本発明の自律型 IC カード 1 は、CPU 10 と、ROM (read only memory) 11 と、RAM (random access memory) 12 と、不揮発性メモリ 13 と、暗号処理部 14 と、通信制御部 15 と、外部通信インターフェース 16 と、ホストデバイスインターフェース 17 とで構成される。

【 0 0 2 4 】

尚、上記構成においては、物理的な構成（ハードウェア）と論理的な構成（ソフトウェア）が混在しているが、以下それぞれについて説明する。

【 0 0 2 5 】

ホストデバイスインターフェース 17 は、従来から備わっている物理的インターフェースであり、外部との通信を行うためのものである。従来と同様、接触型と非接触型がある。CPU 10 は、一部の浮動小数点演算を除き、全ての演算を行う中央演算装置である。尚、浮動小数点演算については別途浮動小数点演算装置が備わっている（図示せず）。ROM 11 は、読み出し専用メモリであり、当該自律型 IC カード 1 用の固有のソフトウェア（OS、プログラム）や後述する本発明チップ ID が格納されている。RAM 12 は、読み書き可能メモリであり、当該自律型 IC カード 1 内で動作するソフトウェアが扱うデータが一時的に記憶されるメモリである。尚、IC カードへの電力供給が絶たれると内容全て消去される。

【 0 0 2 6 】

不揮発性メモリ 13 は、後述する価値情報が格納される。価値情報とは、電子チケット、電子マネー等の価値を含むデータをいう。尚、不揮発性メモリ 13 は、内容保持型メモリなので電力の供給が絶たれても内容が保持される。

以上が物理的な構成部分である。

【 0 0 2 7 】

外部通信インターフェース 16 は、上記 CPU 10 等で構成される後述する本発明チップが外部との直接通信を行う論理的なインターフェースである。従来の IC カードは、それが物理的に接続されている装置、典型的には IC カード端末、とのみ通信を行う論理インターフェースしか持たなかったが、本発明の自律型 IC カード 1 は、論理ホストインターフェースの他に、外部の装置（後述の本発明チップ搭載サービスクライアント）と直接通信を行う外部通信インターフェース 16 を備えている。尚、この外部通信インターフェース 16 は、後の「4. 本発明チップ API プロトコル」で詳述するプロトコルに従って通信を行うインターフェースである。

【 0 0 2 8 】

通信制御部 15 は、外部通信インターフェース 16 を介して外部の装置との間で行われる通信を制御するソフトウェアモジュールである。後述する「10. 本発明チップ API仕様」全体を実現するソフトウェアモジュールであるが、特に「10.1 セッション管理命令群」及び「10.2 トランザクション管理命令群」と関わりが深いソフトウェアモジュールである。このモジュールにより外部の装置との間で直接的に通信が行われる。

【 0 0 2 9 】

暗号処理部 14 は、後述する「3.4 認証・アクセス制御・暗号」、「5. eTP 鍵証明書」及び「6. 鍵実体」において定義された多様な種類の相互認証及び暗号通信を可能にするた

10

20

30

40

50

めのモジュールである。暗号処理部 14 は、通信を行う相手装置に応じて、複数の相互認証及び複数の暗号通信の中から適したものを選択切替している。尚、通信制御部 15 に基づく直接通信の上に、暗号処理部 14 による相互認証及び暗号通信を行えば、間に介在する装置、例えば IC カード端末、によりデータ内容が盗まれたり、データが改竄されるといった危険性が更に低くなる。

【0030】

尚、図 1 に示された本発明に係る構成部は、好適には、最後に詳細にその仕様が列記されるワンチップマイクロコンピュータ（以下、IC チップと称する）で具現化される。

【0031】

ここで、本発明の自律型 IC カードに搭載される IC チップの全体的な特徴を説明する。

10

【0032】

本発明に係る IC チップは、分散環境におけるノードとして機能するように設計されている。従って、本発明の IC カードが IC カードリーダ/ライターを有する IC カード端末本体に接触式又は非接触式で接続されて機能している装置が、ネットワークを介して、例えば他の IC カードと物理的に通信できるように構成されている場合、本発明に係る IC チップを有するそれらの IC カードは通信ネットワーク上でノードとしての機能を有する。そして、このとき IC カードリーダ/ライター及び IC カード端末本体は、ネットワークとコンタクトレス通信の物理層を橋渡しするゲートウェイ（ブリッジ）の役割を果たしている。すなわち、前述の外部通信インターフェース 16 が、分散システムにおけるノードとしての IC カードの通信を司っており、ホストデバイスインターフェース 17 が、コンタクトレス通信の物理層を橋渡しする役割を担っている。

20

【0033】

そこで、分散環境におけるノードとして機能するために、本発明に係る IC チップは、上述のようなネットワークを最も簡単な例とする分散システム全体において固有の識別子（ID）を有している。

【0034】

また、本発明の自律型 IC カードに搭載される IC チップは、ネットワークを介して接続された、同様の IC チップを搭載した複数の装置を識別し、それらの装置との相互認証を先ず行う。そして、それらの装置が正規の実体であると確認した後にそれらの装置との通信を行うように設計されている。その認証においては、2つのモード、すなわち（情報）発行者モード及び所有者モードがある。これは後に詳述する。

30

【0035】

更に、本発明に係る IC チップにおいては、それが有する資源を保護するために、前述の ID に基づいたアクセス制御リストが採用されている。つまり、チップ、ファイル（、レコード）にはアクセス制御リストが付加されている。

【0036】

本発明に係る IC チップでは、アクセス制御リストの中で、「所有者」、「発行者」、「その他」のサービスクライアントを統一的に扱い、それぞれの持つ権限に応じて、アクセス制御リストを変更する API を発行することによって、アクセス権限の制限や解放、移譲といった制御を柔軟に行うことを可能にしている。

40

【0037】

また、本発明に係る IC チップにおいては、コンテンツを複数のコンテンツホルダ、例えば、本発明チップとネットワーク上の価値情報格納サーバー、の間で分散して保持する、ハイブリッド方式をサポートし、そのための機構として、コンテンツ間のリンク機能を提供している。

【0038】

次に、本発明の自律型 IC カードによる通信の具体例について説明する。

図 2 は、本発明の自律型 IC カードによる IC カード間通信を行うシステムの構成を示す図である。

同図において、自律型 IC カード 1a は、IC カード端末本体 3 に備わった IC カードリ

50

ーダ/ライター 2 a に対して接触式又は非接触式に接続されている。また、同様に、自律型 IC カード 1 b は、IC カード 端末本体 3 に備わった IC カードリーダ/ライター 2 b に対して接触式又は非接触式に接続されている。

#### 【0039】

かかる構成において、自律型 IC カード 1 a に対して自律型 IC カード 1 b と通信を行わせたい場合、操作者は、IC カード 端末本体 3 に備えられた入力端末（図示せず）にカード間通信に係る指示を入力する。それに応じて、IC カード 端末本体 3 は、IC カードリーダ/ライター 2 a を介して命令を自律型 IC カード 1 a に送る。その命令を受け取った自律型 IC カード 1 a は、図 1 に示した外部通信インターフェース 1 6 を介して、ソフトウェア的に、直接かつ自律的に、自律型 IC カード 1 b と通信を行う。自律型 IC カード 1 a がこのように自律型 IC カード 1 b と直接通信できるのは、前述のように、自律型 IC カード 1 a が分散環境におけるノードとして機能することができるからである。かかる通信において、図 1 に示した暗号処理部 1 4 により通信データに対して暗号化及び復号化を行えば、通信途中における改竄や盗聴という危険を確実に回避できる。

#### 【0040】

図 3 は、本発明の自律型 IC カードによる通信を行う他のシステムの構成を示す図である。

本発明の自律型 IC カードによる通信においては、通信の相手が IC カードである必要はなく他の通信装置であってもよい。同図において、IC カード 端末本体 3 はネットワーク 4 を介して他の通信装置 5 と接続されている。自律型 IC カード 1 a がネットワーク 4 を介して通信装置 5 と通信を行う場合、IC カード 端末本体 3 から命令を受けた自律型 IC カード 1 a は、ソフトウェア的に、直接かつ自律的に、通信装置 5 と通信を行う。

#### 【0041】

以下、本発明の自律型 IC カードに搭載される IC チップ（以下、単に、本発明チップと称する）の詳細仕様を説明する。以下の説明中、Contents Holder (CHs) が上述の自律型 IC カード 1, 1 a に対応しており、Service Client (SCs) が、自律型 IC カード 1, 1 a と通信を行う全ての装置、すなわち上述の IC カード 端末本体 3、自律型 IC カード 1 b 及び他の通信装置 5 に対応している。

#### 【0042】

尚、全体像を容易に把握できるようにするため、目次を付し、説明本体には各構成ごとに見出しを付けた。

#### 【0043】

目次

### 1. はじめに

#### 1.1 分散環境ノードとしての本発明チップ

#### 1.2 本発明チップ ID で特定する相互認証方式

#### 1.3 本発明チップ ID に基づいたアクセス制御リスト方式による資源保護機構

#### 1.4 チップ所有者・価値情報発行者・価値情報利用者に対する総合的アクセス制御の実現

#### 1.5 ロールバック可能なトランザクション機構

#### 1.6 リンク機能を持った記憶構造

#### 1.7 関連チップ体系との整合性

#### 1.8 多様なチップに対応したゆるやかな標準化

### 2. システム仕様

#### 2.1 システムアーキテクチャ

#### 2.2 本発明チップ識別子（本発明チップ ID）

### 3. 本発明チップ概要

#### 3.1 本発明チップ

#### 3.2 データ構造モデル

#### 3.3 本発明チップ API

#### 3.4 認証・アクセス制御・暗号

10

20

30

40

50

## 4 . 本発明チップ APIプロトコル

### 4.1 パケット形式

### 4.2 命令識別子 ( コマンドID ) 一覧

### 4.3 エラーコード一覧

### 4.4 MAC・トレーラ

### 4.5 セッション通信と非セッション型通信

## 5 . eTP鍵証明書

### セッション構築時における認証時の証明書の利用

#### 1. 概要

#### 6 . 鍵実体

#### 7 . 本発明標準コンテンツ形式

#### 8 . 鍵実体による本発明標準コンテンツ操作時の決算処理

#### 9 . 本発明チップ API仕様 ( データ型定義 )

#### 10 . 本発明チップ API仕様 ( 命令定義 )

##### 10.1 セッション管理命令群

##### 10.2 トランザクション管理命令群

##### 10.3 ファイル管理命令群

##### 10.4 レコード管理命令群

##### 10.5 鍵実体管理命令群

##### 10.6 認証補助管理命令群

#### 暗号実装仕様

##### 1 . はじめに

本発明チップ ( 価値情報付加チップ ) は、本発明関連プロジェクトが目指す、超機能分散システムにおいて、価値情報の格納媒体となるコンピュータシステム ( 例えば IC カード等 ) および、その外部仕様である。今後の技術の進展を想定し、8ビットCPU版、16ビットCPU版、32ビットCPU版とそれぞれシリーズ化されており、共通する操作に関しては共通のコマンド及びメッセージフォーマットを提供する。本書は、特に本発明チップ仕様体系の中でも16ビットCPU版ICチップを対象ハードウェアとした仕様の概要を説明したものである。

#### 【 0 0 4 4 】

本章では、既存の他の耐タンパーチップと比べた場合の、本発明チップの特長を述べる。

#### 【 0 0 4 5 】

##### 1.1 分散環境ノードとしての本発明チップ

本発明チップは従来型のICチップのように、コンピュータの周辺機器として、リーダライタを通して操作されるものではなく、分散環境におけるノードとして設計されている。ネットワーク上のサービス提供モジュールとチップ、チップとカードが対等にpeer-to-peerで通信する。リーダライタ装置は、LANとコンタクトレス通信の物理層を橋渡しするゲートウェイ ( ブリッジ ) となる。

#### 【 0 0 4 6 】

本発明チップアーキテクチャでは、本発明チップは分散システム全体でユニークな識別子 ( 本発明チップID ) を持つ。本発明チップIDは、チップを物理的に識別するだけでなく、分散環境上での経路制御にも利用され、認証通信における相手に識別子として利用される。

#### 【 0 0 4 7 】

従って本発明チップの認証の対象は例えば、リーダライタではなく、ネットワークとリーダライタを経由して、チップと情報交換をするネットワーク上の計算実体 ( Contents Holders ) である。

#### 【 0 0 4 8 】

##### 1.2 本発明チップIDで特定する相互認証方式

本発明チップは、相互認証を行い、正規な実体であることを確認してから通信を行う。本

10

20

30

40

50

発明チップアーキテクチャでは、アクセス対象である本発明チップ搭載Contents Holder (CHs)も、アクセス側である本発明チップ搭載Service Client(SCs)も、ともに一貫したユニークな識別子(本発明チップID)を備え、相互認証後には、相手の本発明チップIDを確実に把握する。

#### 【0049】

#### 1.3 本発明チップIDに基づいたアクセス制御リスト方式による資源確保機構

本発明チップは、相互認証によって本発明チップAPIを発行するSCを特定する。そこで、本発明チップが持つ資源を保護するために、この本発明チップIDに基づいたアクセス制御リストを使う。現在の本発明チップでは、資源に対して、「発行者」(ISSUER)、「所有者」(OWNER)といった属性を、本発明チップIDで表現している。さらに、アクセス制御リストによって、「発行者」・「所有者」・「それ以外」のSCsが発行できる命令を指定

10

#### 【0050】

1.4 チップ所有者・価値情報発行者・価値情報利用者に対する統合的アクセス制御の実現  
本発明チップでは、価値情報を格納する多様なアプリケーションが実装される可能性がある。価値情報には様々なタイプがあり、例えば、チップ上にある情報でも、以下のようなものが考えられる。

#### 【0051】

- チップの所有者は変更できずに情報の発行者だけが変更できる情報  
(例：電子チケットの座席番号)
- チップの所有者でさえ見せない情報(例：電子チケット変更の鍵)
- チップの所有者だけが完全に制御できる情報(例：所有者の個人情報)
- 読むことはだれでもできる情報

20

本発明チップでは、アクセス制御リストの中で、「所有者」、「発行者」、「その他」のSCsを統一的に扱い、それぞれのもつ権限に応じて、アクセス制御リストを変更するAPIを発行することによって、アクセス権限の制御や解放、委譲といった制御を柔軟に行うことを可能にしている。

#### 【0052】

#### 1.5 ロールバック可能なトランザクション機構

本発明チップに価値情報を移動する時は、安全に行う必要がある。そこで、本発明チップでは、特に価値情報の作成と削除に関しては、処理のAtomicityを保証するために、トランザクション機構を提供している。トランザクションを開始したあと、発行された命令による操作は、トランザクション終了時のコミット命令によって反映される。アポート命令が発行された場合、また規定したタイムアウト時間に達するまでコミット命令が到着しない場合は、発行された命令に関してロールバック(roll-back)される。同様に、トランザクション途中で、何らかのトラブルによって、本発明チップが電源遮断された場合は、次に本発明チップがアクティブになった初期時にロールバック処理が行われる。

30

#### 【0053】

#### 1.6 リンク機能を持った記憶構造

現在、耐タンパーチップは、情報処理能力においても記憶容量などの資源においても多様なものが存在する。その中でも、資源に乏しいハードウェアでは、想定するアプリケーションの全ての価値情報をチップに格納できないケースもある。一方、チップがそれぞれ分散して価値情報を保持する方式も、処理効率の観点から望ましい。そこで、本発明ではコンテンツを複数の本発明コンテンツホルダ、例えば、本発明チップとネットワーク上の価値情報格納サーバー、の間で分散して保持する、ハイブリッド方式をサポートし、そのための機構として、コンテンツ間のリンク機能を提供している。

40

#### 【0054】

#### 1.7 関連チップ体系との整合性

本発明チップはAPIのコマンドコンベンション、エラーコード、資源の静的・動的な扱い方の方法などを本発明関連プロジェクトの他のアーキテクチャと整合性を持たせている。

50

例えば、エラーコードの一般的な部分は関連チップと本発明チップは共通である。これによって、関連チップアーキテクチャ上での開発経験のある技術者が本発明チップアプリケーションを開発することを支援する。

【 0 0 5 5 】

#### 1.8 多様なチップに対応したゆるやかな標準化

本発明チップは、8bit CPU ~ 32bit CPU、Contact/Contact-less/Dual Interface、スマートカード ~ RF IDと、様々なICカードに対して、整合性のある一貫したAPIの体系を提供する。

【 0 0 5 6 】

### 2 . システム仕様

10

#### 2.1 システムアーキテクチャ

本発明チップアーキテクチャは、耐タンパ性を有する価値情報を安全に格納し、コンピュータネットワーク上を安全に交換するための分散システムアーキテクチャであり、主に、以下の要素から構成される。

【 0 0 5 7 】

#### (A) 価値情報ネットワーク系 (Entity Network Infrastructure)

価値情報を交換するためのアーキテクチャ。以下の二つの要素から構成される。

【 0 0 5 8 】

##### A-i) 本発明チップ搭載コンテンツホルダ (CHs)

本発明チップ搭載コンテンツホルダは、価値情報を格納し、本発明チップ搭載 APIを提供してこの価値情報を外部から操作させる、分散システム上の計算実体で。例として、以下のようなものがある。

20

【 0 0 5 9 】

本発明チップ：

価値情報を安全に格納する、耐タンパ性を有する L S I チップである。

【 0 0 6 0 】

本発明チップボックス：

価値情報を安全に格納する、耐タンパをもった筐体に収められた大容量電子金庫。

【 0 0 6 1 】

##### A-ii) 本発明チップ搭載サービスクライアント (SCs)

30

本発明チップ搭載コンテンツホルダに格納された価値情報に本発明チップ APIを介してする計算実体。例えば、以下のようなタイプがある。

【 0 0 6 2 】

本発明チップ：

本発明チップ自体も高度なものはクライアントとしての機能ももつ。

【 0 0 6 3 】

発行サーバ：

価値情報を発行し本発明チップ搭載コンテンツホルダに格納する。

【 0 0 6 4 】

サービス現場システム：

40

本発明チップに格納された価値情報を用いたアプリケーションシステム。電子チケットのゲートなどが含まれる。

【 0 0 6 5 】

本発明チップ搭載サービスクライアント且つ本発明チップボックスである計算実体も存在することに注意。

【 0 0 6 6 】

#### (B) 暗号 / 認証ネットワーク系 (AENI: Authentication/Encryption Network Infrastructure)

本発明チップが公開鍵暗号系の認証及び暗号を実装した場合の、認証系。認証局が主な役割となる

50

(C) アプリケーションネットワーク系 (ANI:Application Network Infrastructure) アプリケーションに依存した各種通信系。例えば、電子チケットであれば、チケットの検索や購入を指示するネットワークプロトコルなどが含まれる。この購入の指示までがANIの枠組みで行われ、その後、価値情報であるチケット本体の移動は、本発明チップの機能を使ったENIで行われる。

【0067】

2.2 本発明チップ識別子 (本発明チップID)

本発明チップ搭載コンテンツホルダとサービスクライアントは、ユニークな識別子を持ち、それぞれ本発明チップ識別子 (本発明チップID) と呼ぶ。本発明チップ識別子は、ネットワーク上で、本発明チップコンテンツホルダやサービスクライアントの認証、メッセージの経路制御などに用いられる。本発明チップIDは、16オクテット (128ビット) 数で表現される。

10

【0068】

本発明チップIDは、ENIでの通信プロトコルで用いられる識別子の交換フォーマットであり、その本発明チップ内での内部保存形式を定めるものではない。

【0069】

経路制御の実装の容易性を考慮すると、ENIのネットワーク層のプロトコルにおける識別子を直接本発明チップIDとする実装方法も考えられる。たとえば、本発明チップIDの下位ビットがそのままネットワークアドレスであるなど。

【0070】

所有者の本発明チップIDは、「0x00~00」(all"0")とする。

20

【0071】

例えば、Xという本発明チップIDをもった本発明チップ搭載サービスクライアントが所有者認証モードで認証されてセッションをはり、そのセッションの中でファイルを作ったとき、そのファイルの発行者ID欄は「0」が格納されるものとする。

【0072】

管理運用のスーパーユーザの本発明チップIDは「0xff~ff」(all"1")とする。

【0073】

3. 本発明チップ概要

30

3.1 本発明チップ

本発明チップは、価値情報を用いた各種会社サービスを直接享受する利用者が価値情報を携帯するために用いる小型計算実体であり、本仕様書が主に扱う対象である。通常、ICカード、スマートカード、携帯型端末といったハードウェア上に実装されることが多いと考えられている。これらのハードウェアには、備えている情報処理能力や資源の差が大きいこと、耐タンパーチップ技術が発展段階であることを考慮し、本発明チップの仕様は、様々なハードウェアに応じた、多様な仕様を提供し、インタフェースのみを共通化し、シリーズ化を図る。

【0074】

【表1】

40

本発明チップ/8	8ビットCPUチップ用
本発明チップ/16	16ビットCPUチップ用
本発明チップ/32	32ビットCPUチップ用
本発明チップ/T	端末(Terminal)用

表：本発明チップ仕様のシリーズ

3.2 データ構造モデル

50

本発明チップボックスや本発明チップに格納される価値情報は、次の階層的なデータ構造として外部から見えるものとする。

【 0 0 7 5 】

フォルダ

複数のファイルのセットをまとめる構造。

【 0 0 7 6 】

本発明チップ / 1 6 にルートフォルダだけが存在する。

【 0 0 7 7 】

ファイル

価値情報を格納する資源。ファイルの中にはレコードが格納されるが、本発明チップ / 1 6 では、一つのファイルに単一のレコードだけが格納される構造をとる。

【 0 0 7 8 】

### 3.3 本発明チップ API

本発明チップは、本発明チップ搭載サービスクライアントから、本発明チップ API によって動作する。

【 0 0 7 9 】

本発明チップ API は、基本的に、以下の手順によるセッションベースのプロトコルである。

【 0 0 8 0 】

- 本発明チップと本発明チップ搭載サービスクライアントの相互認証と暗号通信路の確立  
セッション (トランザクション) の確立

- 命令メッセージの送付

- セッション (トランザクション) の切断

セッション / トランザクション方式はメッセージ交換回数が多くなるため、タッチ&ゴーのような高応答性が求められる応用に適さないことがある。そこで、命令メッセージ毎に認証や暗号を適用する。非セッションプロトコルも一部の命令に関して取り入れる。

【 0 0 8 1 】

session ID = 0 で発行した命令は、非セッションモードとして扱われる。

【 0 0 8 2 】

セッション命令

eopn\_ses Open Session

ecfm\_ses Confirm Session

ecls\_ses Close Session

トランザクション命令

eopn\_tra Open Transaction

ecfm\_tra Confirm Transaction

ecom\_tra Commit Transaction

eabo\_tra Abort Transaction

ファイル操作命令

ecre\_fil Create File

edel\_fil Delete File

etra\_fil Transfer File

eupd\_fim Update File Mode

eenc\_fil Encode File

edec\_fil Decode File

レコード操作命令

eupd\_rec Update Record

erec\_rec Read Record

鍵実体操作命令

ecre\_key Create Key

10

20

30

40

50

edel\_key Delete Key  
 eupd\_key Update Key  
 認証補助命令  
 ecfm\_cer Confirm Certificate  
 制御命令  
 epol\_chp Polling Chip  
 eini\_car Initialize Card  
 eupd\_cer Update My Certificate  
 eupd\_cpk Update CA Public Key

### 3.4 認証・アクセス制御・暗号

10

セッションを構築するとき、本発明チップと本発明チップ搭載サービスクライアントが相互認証を行う。この認証の具体的アルゴリズムは、チップに搭載されているハードウェア等に特化して、多様なものがあるものとする。

#### 【0083】

##### 認証モード

認証には（情報）発行者モードと所有者モードのモードがあり、認証時に指定され、各モードによって認証アルゴリズムが（一般的には）異なる。

#### 【0084】

##### （情報）発行者モード：

本発明チップ搭載サービスクライアントをファイルの発行者として認証するモード。発行者のモードで認証された後は、その本発明チップサービスクライアントが作成したファイルには、発行者権限でアクセスでき、それ以外の資源には、その他権限でアクセスすることができる。

20

#### 【0085】

##### 所有者のモード：

本発明チップ搭載サービスクライアントをチップの所有者として認証するモード。通常、パスワード等、人間にとって扱いやすい認証方法が使われる。所有者モードで認証された本発明チップ搭載サービスクライアントは所有者権限を持つ。

#### 【0086】

##### 相互認証後の本発明チップの状態

認証が終了すると、本発明チップは、以下の情報を保持することになる。

30

#### 【0087】

- 正規に認証された本発明チップ搭載サービスクライアントの本発明チップID
- 本発明チップ搭載サービスクライアントへ送るメッセージの暗号共通鍵
- 本発明チップ搭載サービスクライアントから来たメッセージの復号共通鍵
- セッションID
- セッションモード（発行者モード/所有者モード）

##### アクセス制御

チップ、ファイル（、レコード）には、

- 発行者を表す本発明チップID
- アクセス制御リスト（権限毎に許可されている操作）

40

がつけられており、以下で定めるセッションがもつ権限に応じて、アクセスが制限される。

#### 【0088】

##### ファイルアクセスモード

本発明チップID=eidのSCが、本発明チップ内のファイルF（Fを作成した本発明チップSCの本発明チップID=F.eid）をアクセスするモードには、以下の3種類がある。

#### 【0089】

##### 1. 所有者アクセス

所有者認証を通過してセッション中でのアクセス。

50

【 0 0 9 0 】

この場合、eid=0x00(all"0)である。

【 0 0 9 1 】

2. (情報) 発行者アクセス

(情報) 発行者認証を通過したセッション中から、eid==F.eidの場合のアクセス。

【 0 0 9 2 】

3. その他のアクセス

(情報) 作成者認証を通過したセッション中で、eid !=F.eidの場合のアクセス、

もしくは、

非セッションモードによるアクセス。

【 0 0 9 3 】

ルートフォルダへのアクセス制御リスト

【表 2】

F	E	D	C	B	A	9	8
予約	予約	予約	予約	予約	予約	予約	予約
7	6	5	4	3	2	1	0
予約	予約	ACL5	ACL4	ACL3	ACL2	ACL1	ACL0

ACL0 =0 / 1 ecre\_fil所有者アクセス不可 / 可

所有者アクセスで、ルートフォルダーにファイルを作成することを許可するか

ACL1 =0 / 1 ecre\_filその他アクセス不可 / 可

所有者アクセス以外で、ルートフォルダーにファイルを作成することを許可するかどうか

ACL2 =0 / 1 edel\_fil所有者アクセス不可 / 可

所有者アクセスで、ルートフォルダーからファイルを消すことを許可するか

ACL3 =0 / 1 edel\_filその他アクセス不可 / 可

所有者アクセス以外で、ルートフォルダからファイルを消すことを許可するかどうか

ACL4 =0 / 1 etra\_fil所有者アクセス不可 / 可

所有者アクセスで、ルートフォルダーからファイルを転送することを許可するかどうか

ACL5 =0 / 1 etra\_filその他アクセス不可 / 可

所有者アクセス以外で、ルートフォルダからファイルを転送することを許可するかどうか

【表 3】

	cre_fil	del_fil	tra_fil
所有者アクセス	ACL0	ACL2	ACL4
発行者アクセス	-	-	-
その他アクセス	ACL1	ACL3	ACL5

10

20

30

40

50

ルートフォルダの（情報）発行者は所有者として運用する。

【 0 0 9 4 】

つまり、価値情報をルートフォルダにファイル/鍵実体を作成する一般の価値情報サーバは、ルートフォルダによっては、「その他アクセス」に相当するファイルへのアクセス制御リスト

【表 4】

F	E	D	C	B	A	9	8
ACLf	ACLe	ACLd	ACLc	ACLb	ACLa	ACL9	ACL8
7	6	5	4	3	2	1	0
ACL7	ACL6	ACL5	ACL4	ACL3	ACL2	ACL1	ACL0

10

ACL0 =0 / 1 eupd\_rec所有者アクセス不可 / 可  
 ACL1 =0 / 1 eupd\_recその他アクセス不可 / 可  
 ACL2 =0 / 1 erea\_rec所有者アクセス不可 / 可  
 ACL3 =0 / 1 erea\_recその他アクセス不可 / 可  
 ACL4 =0 / 1 eupd\_fim所有者アクセス不可 / 可  
 ACL5 =0 / 1 eupd\_fimその他アクセス不可 / 可  
 ACL6 =0 / 1 elst\_fil所有者アクセス不可 / 可  
 ACL7 =0 / 1 elst\_filその他アクセス不可 / 可  
 ACL8 =0 / 1 edel\_fil所有者アクセス不可 / 可  
 ACL9 =0 / 1 edel\_filその他アクセス不可 / 可  
 ACLa =0 / 1 etra\_fil所有者アクセス不可 / 可  
 ACLb =0 / 1 etra\_filその他アクセス不可 / 可  
 ACLc =0 / 1 eenc\_fil所有者アクセス不可 / 可  
 ACLd =0 / 1 eenc\_filその他アクセス不可 / 可  
 ACLe =0 / 1 edec\_fil所有者アクセス不可 / 可  
 ACLf =0 / 1 edec\_filその他アクセス不可 / 可

20

30

【表 5】

	upd_rec	rea_rec	upd_fim	lst_fil	del_fil	tra_fil	enc_fil	dec_fil
所有者 アクセス	ACL0	ACL2	ACL4	ACL6	ACL8	ACLa	ACLc	ACLe
発行者 アクセス	常に可							
その他 アクセス	ACL1	ACL3	ACL5	ACL7	ACL9	ACLb	ACLd	ACLf

40

鍵実体のアクセス制御リスト

【表 6】

F	E	D	C	B	A	9	8
ACLf	ACLe	ACLd	ACLc	ACLb	ACLa	ACL9	ACL8
7	6	5	4	3	2	1	0
ACL7	ACL6	ACL5	ACL4	ACL3	ACL2	ACL1	ACL0

ACL0 =0 / 1 eupd\_key所有者アクセス不可 / 可  
 ACL1 =0 / 1 eupd\_keyその他アクセス不可 / 可  
 ACL2 =0 / 1 erea\_key所有者アクセス不可 / 可  
 ACL3 =0 / 1 erea\_keyその他アクセス不可 / 可  
 ACL4 =0 / 1 eupd\_kym所有者アクセス不可 / 可  
 ACL5 =0 / 1 eupd\_fimその他アクセス不可 / 可  
 ACL8 =0 / 1 edel\_key所有者アクセス不可 / 可  
 ACL9 =0 / 1 edel\_keyその他アクセス不可 / 可  
 ACLc =0 / 1 eenc\_fil所有者アクセス不可 / 可  
 ACLd =0 / 1 eenc\_filその他アクセス不可 / 可  
 ACLe =0 / 1 edec\_fil所有者アクセス不可 / 可  
 ACLf =0 / 1 edec\_filその他アクセス不可 / 可

【表 7】

	upd_key	rea_key	upd_kym	del_key	enc_fil	dec_fil
所有者アクセス	ACL0	ACL2	ACL4	ACL8	ACLc	ACLe
発行者アクセス	常に可	常に可	常に可	常に可	常に可	常に可
その他アクセス	ACL1	ACL3	ACL5	ACL9	ACLd	ACLf

#### 4 . 本発明チップ APIプロトコル

##### 4.1 パケット形式

パケット形式右に、「 」は暗号セッション時に、非暗号化部分、「 」は暗号化する部分、

2 オクテット以上の数値データは、Little Endianで格納する。

【 0 0 9 5 】

ルーティング・ヘッダ

【表 8】

10

20

30

40

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7	
本発明チップ ID (Destination 0)								<input type="checkbox"/>
本発明チップ ID (Destination 1)								<input type="checkbox"/>
本発明チップ ID (Source 0)								<input type="checkbox"/>
本発明チップ ID (Source 1)								<input type="checkbox"/>

10

本発明チップセッション部 (往)

【表 9】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7	
SID	Serial No.	Command ID		データ長 (バイト数)		予約		<input type="checkbox"/> <input type="checkbox"/>
DATA								■
.....								■
DATA								■

20

本発明チップセッション部 (復)

【表 10】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7	
SID	Serial No.	Error Code		データ長 (バイト数)		予約		<input type="checkbox"/> <input type="checkbox"/>
DATA								■
.....								■
DATA								■

30

MAC・トレーラ

【表 11】

40

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7	
MAC ID		MAC Length		予約				<input type="checkbox"/>
MAC(1)								<input type="checkbox"/>
MAC(MD5 例:2)								<input type="checkbox"/>

MACは、暗号化された後のデータに対してつけられる。

10

【 0 0 9 6 】

#### 4.2 命令識別子 (コマンドID) 一覧

本発明チップ APIの命令識別子 (コマンドID) は正の1バイト整数で表現される。

【 0 0 9 7 】

本発明チップ / 8 , 1 6 , 3 2 のシリーズで共通に定められており、以下の通りである。

【 0 0 9 8 】

【 表 1 2 】

ID	ニモニック	8/16/32	意味
0x01	eopn ses	○○○	Open Session
0x02	ecfm ses	○○○	Confirm Session
0x03	ecls ses	○○○	Close Session
0x04	eopn tra	×○○	Open Transaction
0x05	ecfm tra	×○○	Confirm Transaction
0x06	ecom tra	×○○	Commit Transaction
0x07	eabo tra	×○○	Abort Transaction
(0x11	ecre fol	××○	Create Folder)
(0x12	edel fol	××○	Delete Folder)
(0x13	eupd fom	××○	Update Folder Mode)
0x21	ecre fil	○○○	Create File
0x22	edel fil	○○○	Delete File
0x23	eira fil	×○○	Transfer File
(0x24	erdm fil	××○	Redeem File)
0x25	eenc fil	×○○	Encode File
0x26	edec fil	×○○	Decode File
0x27	eupd fim	○○○	Update File Mode
0x2F	elst fid	○○○	List File ID
(0x31	ecre rec	××○	Create Record)
(0x32	edel rec	××○	Delete Record)
0x33	eupd rec	○○○	Update Record
0x34	erea rec	○○○	Read Record
(0x35	eupd rem	××○	Update Record Mode)
0x41	epol car	○○○	Poll Card
0x51	eini car	○○○	Initialize Card
0x52	eupd cer	×○○	Update My Certificate
0x53	eupd cpk	×○○	Update CA Public Key
0x61	ecre key	×○○	Create Key
0x62	edel key	×○○	Delete Key
0x63	eupd key	×○○	Update Key
0x71	ecfm cer	×○○	Confirm Certificate

10

20

30

表：本発明チップ APIの命令識別子一覧

本発明チップ / 16 第一バージョンでは実装せず。

【0099】

#### 4.3 エラーコード一覧

本発明チップにおける命令のエラーコードは、符号付き整数で表される。操作が正常に終了したときは、E\_OK = 0 または、操作の結果を表す正の数が返される。正常に終了せず、何かのエラーがあった場合には、負の数を返す。エラーコードは以下のコンベンションで表現される。

【0100】

・"E\_"と書かれたものは、チップ中のローカルな命令処理時に起きる一般的なエラーを表す(関連チップのエラーコードと共通)。

【0101】

・"EN\_"と書かれたものは、通信関連の一般的なエラーを表す(セッションエラー以外は、関連チップのエラーコードと共通)。

40

50

## 【 0 1 0 2 】

・"EE\_"と書かれたものは、主に、セキュリティー関連の一般的なエラーを表す（本発明チップ専用）。

## 【 0 1 0 3 】

## 【表 1 3】

コード	ニモニック	意味
-0x00	E_OK	正常終了
-0x05	E_SYS	システムエラー
-0x0a	E_NOMEN	メモリ不足
-0x11	E_NOSPT	未サポート機能
-0x21	E_PAR	パラメーターエラー
-0x23	E_ID	不正ID番号
-0x34	E_NOEXS	オブジェクトが存在しない
-0x3f	E_OBJ	オブジェクトの状態が不正
-0x41	E_MACV	メモリアクセス不能、メモリアクセス権違反
-0x42	E_OACV	オブジェクトアクセス権違反
-0x55	E_TMOUT	ポーリング失敗またはタイムアウト
-0x72	EN_OBJNO	本発明チップがアクセスできないオブジェクト番号指定
-0x73	E_PROTO	本発明チップがサポートしていないプロトコル
-0x74	E_RSFN	本発明チップがサポートしていない命令
-0x77	EN_PAR	本発明チップがサポートしていない範囲のパラメータを指定した
-0x7a	EN_EXEC	本発明チップ側の資源不足のため実行できない
-0x7f	EN_NOSES	指定したセッションがない
-0x81	ES_AUTH	認証エラー
-0x82	ES_EACL	アクセス制御違反

表：本発明チップ APIのエラーコード一覧

## 4.4 MAC・トレーラ

## 【表 1 4】

10

20

30

40

MAC ID	Length (bytes)	内容
0x01	16	Check SUM
0x02	2	CRC-CCITT
0x03	4	予約
0x04	16	予約
0x05	16	HMAC with MD5 (RFC 2085)
0x06	16	予約

10

### 1 チェックサム

【表 1 5】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
MAC ID		MAC Length		予約			
MAC (1)							
.....							
MAC (16)							

20

### 2 CRC-CCITT

30

CRC-CCITTにおける生成多項式

$$G(X) = X^{16} + X^{12} + X^5 + 1$$

パケットサイズが奇数の場合、後ろのバイトをnull paddingしてからCRCをとる。

【0104】

【表 1 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
MAC ID		MAC Length		予約			
MAC							

40

### 5 HMAC with MD5 [RFC 2085]

セッション/トランザクション暗号通信路で用いる

暗号通信路の秘密鍵暗号の鍵の始めの64バイトを鍵Kとして用いる、

HMACは以下の式で計算する。

【0105】

$$MD5(K \text{ xor } opad \parallel MD5(K \text{ xor } ipad \parallel \text{TEXT}))$$

opad+the byte 0x36 repeated 64 times

ipad+the byte 0x5c repeated 64 times

50

" | "means simple bit append.  
 e.g. ("0100" | "1001")=("01001001")

セッション確立後、暗号通信を行なっている場合、64バイトの秘密鍵として、16バイト暗号鍵を4回繰り返したもの、つまり、eopn\_ses( )のRSA認証による場合、交換された乱数を以下のようにアペンドしてを用いる。

【0106】

$K = R a | R b | R a | R b | R a | R b | R a | R b$

【表17】

D0	D1	D2	D3	D4	D5	D6	D7
MAC ID		MAC Length		予約			
MAC (1)							
MAC (2)							

10

4.5 セッション通信と非セッション型通信

本発明チップとの通信では、「相互認証」後に「暗号」通信路としてのセッション/トランザクションが確立し、以後「暗号」化されたメッセージ通信を行う。erea\_rec命令(レコードの内容を読む)に関しては、「認証」のフェーズを経ずに、直接実行する非セッション型通信をサポートする。この時、セッションを張らずに命令をsession ID=0で呼び出すことで起動できる。この非セッション型通信によるアクセスは、そのファイルが「その他アクセス」を許可していれば成功する。この時、アクセスパケットは暗号化されない。これは、チケットのゲーティングなどの、タッチ&ゴー応用の必要性が高い場所で用いられる。

20

【0107】

サービス提供者は、このモードを使うことにより、リスクは高くなる。以下の音源チップを搭載しています。

【0108】

30

5 . eTP鍵証明書

セッション構築時における認証時の証明書の利用

1. 概要

本発明チップ/16ではセッションを構築する時に相互認証は、公開鍵ベースかつ、PKIの存在を前提とする。

【0109】

- 本発明チップ搭載CHs、本発明チップ搭載SCsには、それぞれ本発明チップIDが割り振られ、発行時に、認証用の公開鍵/秘密鍵の対が割り当てられている。

【0110】

- セッション・トランザクション構築時に、正規に発行された(本発明チップID、公開鍵)対を持っていることを相互に示すことができるように、常に本発明チップ搭載CH、本発明チップ搭載SC内部に本発明チップ搭載CAの署名付の証明書を格納している。

40

【0111】

本発明チップ搭載CAは、将来的には複数あることが想定されるが、本バージョンでは一つに制限する。

【0112】

- 証明書は、本発明チップID、公開鍵のペア、(id,PK\_id)が正規に発行されたものであることを、本発明チップ搭載CAの署名(S\_ca(id,PK\_id))が証明する。

【0113】

2. 詳細

50

本発明チップ搭載ノード（ここで、本発明チップ搭載SCと本発明チップ搭載CH両方を総称してそう呼ぶ）は、認証段階の前段階として、相手のデバイスとの間で、相互に自らの本発明チップの公開鍵を本発明チップデバイス発行者の署名つき証明書として送付する（注1）。

【0114】

本発明チップ搭載ノードは、証明書発行者の署名鍵の公開鍵を証明書形式で所有する。この鍵を使って正当性を確認する。

【0115】

「相手ノードから送られてきた公開鍵の正当性」  
= 「発行者の正規な署名があること」 + 「証明書が有効期間内であること」  
を検証する（注2）。

10

【0116】

正当であることが確認できない場合にはセッションを確立せずに、通信を打ち切る（注3）。

【0117】

注1）

公開鍵ベースの認証において、その後のセッションで、高価な価値の転送/交換の場合には、証明書が「廃止・破棄」されている場合も考慮する。本発明チップ/16では、

1. CRL(破棄リスト)を使う（ただしデータが大きくなる）

または

2. 特定の証明書が「廃止・破棄」されているかを照会するサーバーアクセスの利用を行なう。

20

【0118】

注2）

注1で述べた用に、これに加えて証明書の「廃止・破棄」の問い合わせを行なうこともできる。

【0119】

注3）

証明書の正当性の確認が終了する前に、とりあえず認証のステップを開始して、相手に認証に必要なパケットを送信している間、あるいは相手からの認証に必要なパケットを待っている間に別タスクによつての正当性を確認する実装が有効である。チップ内で割り込みやマルチタスクOSが実現されている必要があるが、認証全体の処理時間を短く実装するために有効な手段である。

30

【0120】

上の説明は、本発明チップやカードのようなノードの通信の初期の設定にかかる時間の減らす工夫である。サーバーマシン上に実装されたノードは、不正な証明書をもつ装置から不正な証明書の送付によるサーバ問い合わせ処理の浪費というDoSを防ぐため、必ず最初に証明書の署名確認をオフラインで済ませてから「廃止・破棄」を確認した方が良い。

【0121】

eTP鍵証明書のフォーマット(ver.0.1)

40

【表18】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
ver	予約	証明書番号		署名法	予約	発行者 ID	
有効期間開始				有効期間終了			
本発明チップ ID (1)							
本発明チップ ID (2)							
CAKEY#	予約	キー長さ		署名長さ		予約	
公開鍵(1)							
.....							
公開鍵(17)							
署名(1)							
.....							
署名(16)							

10

20

・ ver :

証明書形式のバージョン番号。ver.0.1ではver="1"とする。

【 0 1 2 2 】

・ 証明書番号 :

その本発明チップIDに対する鍵の証明書シリアル番号。

【 0 1 2 3 】

・ 署名法 :

署名アルゴリズム指定子

0 : 証明書のkeyフィールド部分までをSHA-1ハッシュし、結果をRSA暗号化 (実装)

1 : 予約

2 : 予約

3 : 予約

・ 発行者ID :

証明書の発行者 (CA局) ID。ver.0.1では"0"に固定。

【 0 1 2 4 】

・ 有効期間開始、有効期限終了 :

証明書の有効期間。

【 0 1 2 5 】

符号なし4オクテット整数 (2000年1月1日AM0:00からの秒数の積算)

・ C A K E Y # :

CAが鍵の署名に複数の鍵を使っている場合に、それを区別するための鍵のID番号。ver.0.1では"0"に固定。

【 0 1 2 6 】

・ キー長さ :

後ろのつく鍵のビット長サイズ

証明書中で、鍵が8オクテットの倍数にならない場合、それを格納するkeyフィールドの最後に"0"を付加し (null padding) 8オクテットの倍数データとして格納する。key

30

40

50

フィールドの中の「キー長さ」ビットが有効な鍵である。

【 0 1 2 7 】

・署名長：

最後につく署名に使うハッシュ (sha-1 など) のビット長

ハッシュは、必要であれば最後に null padding して 8 オクテットの倍数のデータとしてから、CAの鍵で暗号化して署名を作成する。

【 0 1 2 8 】

署名を確認する時には、最後につくこの 8 オクテットの整数倍のデータを、本発明チップ搭載ノードが保存するCAサーバーの公開鍵で復号する。その復号値の中で、先頭の署名長ビットが確認に用いるハッシュ値となる。

【 0 1 2 9 】

・具体的な長さの例：(RSA/sha-1署名を使った場合)

- R S A

本発明チップ認証公開鍵はRSAで1024ビットとする。

【 0 1 3 0 】

1024bits = 128bytes = 16 × 8bytesである。

【 0 1 3 1 】

sha-1を署名のためのハッシュ関数として使う。

【 0 1 3 2 】

- sha-1

sha-1の出力は160bits.

160bits=20bytes.

このバイト長を null padding して 8 の倍数にして 2 4 バイトデータをRSA暗号で署名する。

【 0 1 3 3 】

証明書全体の長さは以下のように計算できる。

【 0 1 3 4 】

**証明書の長さは**

・固定長部分 (40バイト)

**鍵の部分の部分の長さ**

・署名部分 (128バイト)

**の和 (168バイト) になる。**

【 0 1 3 5 】

署名の部分は暗号結果は1024ビットを必要とするので、128バイトの長さを使う。

【 0 1 3 6 】

ただし、鍵の部分は単純に128バイトとはならないので注意されたい。

【 0 1 3 7 】

これはRSA鍵の場合、公開鍵として1024ビットと俗称している場合には1024bitsのNと指数e ( e , N ) の対を念頭においており、鍵の部分には ( e , N ) 両方を入れる必要があるためである。

【 0 1 3 8 】

RSAの場合、鍵フィールドには以下のようにe,Nを入れることにする。

【 0 1 3 9 】

eのバイト長：( 1 バイト )

eの値そのもの：必要に応じてバイト単位まで広げてバイト列として埋める。これに続けてNの値をバイト列として埋める。

【 0 1 4 0 】

10

20

30

40

50

( e , N ) で e = 3 の場合を例として考えてみると、鍵のフィールドには以下のように130バイトのバイトデータが入る。

【 0 1 4 1 】

公開鍵 = 0x01 | 0x03 | N のバイト列 ( 128バイト )

130は、8の倍数でないので、136バイトまでパディングされて鍵データの占めるバイトは136となる。

【 0 1 4 2 】

これで全体長が計算できる。

【 0 1 4 3 】

固定長部分 ( 40バイト)  
 + 鍵の部分の占める長さ (136バイト)  
 + 署名部分 (128バイト)  
 = 304バイト

10

このように e = 3 の場合には、長さは総計で304バイトとなる。

【 0 1 4 4 】

#### 6 . 鍵実体

鍵実体は、本発明チップが通常のファイルに格納する、暗号鍵及び暗号鍵を使った価値情報操作に必要な情報をまとめたシステムデータ形式である。

20

【 0 1 4 5 】

【表 1 9】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
keyObj ID		暗号方式指定子		予約			
KEY (1)							
KEY (2)							
BANK ID		予約		支払残高			
LEN				予約			

30

暗号方式指定子：

KEY (1)、KEY (2) が、どの暗号アルゴリズムの鍵であるかを指定する。以下の通り、eopn\_sesにおける暗号通信路の暗号アルゴリズム指定と同じ方式で指定する。

【 0 1 4 6 】

【表 2 0】

40

暗号方式指定子	暗号種類
0x01	DES
0x02	3DES
0x03	Rijndael
0x04	Hierocrypt-3
0x05	Camellia

10

### 7. 本発明標準コンテンツ形式

標準コンテンツ形式は、本発明チップ内で価値情報操作を行なうことができる。標準コンテンツデータ形式である。鍵実体を操作することによって、暗号/復号操作を加え、その際に鍵実体の内部の課金情報をアトミックに変更することができる。

【0147】

基本的に、暗号化されたコンテンツを復号化する時に課金するという販売モデムを本発明チップの内部で確実にこなうことを想定している。

20

【0148】

コンテンツ形式全体

【表21】

(A) コンテンツヘッダ
(B) 課金ヘッダ
(C) 暗号化コンテンツ本体
(D) 署名トレーラ

30

(A) コンテンツヘッダ

【表22】

D0	D1	D2	D3	D4	D5	D6	D7
Contents ID		Segment ID		Publisher Info.			

40

(B) 課金ヘッダ

【表23】

D0	D1	D2	D3	D4	D5	D6	D7
Payment Scheme		表現式長		Parameter(1)		Parameter(2)	
Parameter(3)		Parameter(4)		Parameter(5)		Parameter(6)	

50

## (C) 暗号化コンテンツ

【表 2 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
暗号方式指定子		データ長LEN		予約			
暗号化KEY(1)							
暗号化KEY(2)							
暗号化コンテンツ本体(1)							
.....							
暗号化コンテンツ本体(m)							

10

暗号化KEY(1),(2)は、そのコンテンツの解読に必要な鍵実体に格納されている鍵を使って暗号化されている。本発明チップ/16ではファイル容量の問題から、鍵実体の暗号として、共通鍵暗号だけを考え、公開鍵暗号は扱わないものとする。(ここの暗号化キーのデータ長が暗号後に、伸長してしまうため)

20

## (D) 署名トレーラ

【表 2 5】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
署名ID		署名長		予約			
署名(1)							
.....							
署名(n)							

30

署名の方式は、eTPパケットの署名トレーラと同じ形式とする。

## 【0149】

HMAC with MD5 [RFC 2085] の場合の64バイトの秘密鍵として、本コンテンツに含まれている暗号化KEY(1)と暗号化KEY(2)を、鍵実身の鍵で復号したK1、K2を使い、Kを以下の方法で生成する。

## 【0150】

$$K = K1 | K2 | K1 | K2 | K1 | K2 | K1 | K2$$

課金ヘッダ

40

【表 2 6】

支払方式	Payment schemeParameter 1	Parameter 2	...
固定価格支払	0x01	価格(単位: 銭)	-

50

## 暗号化コンテンツ

【表 2 7】

暗号方式指定子	暗号種類
0x01	DES
0x02	3DES
0x03	Rijndael
0x04	Hierocrypt-3
0x05	Camellia

10

## 署名トレーラ

本発明チップのパケットと同様の仕様

8. 鍵実体による本発明標準コンテンツ操作時の決算処理

鍵実体を使った計算方法は、以下の方式をとる。

## 【 0 1 5 1 】

20

( 1 ) 本発明暗号化コンテンツの課金ヘッダの課金額 ( 本発明チップ/16では、固定額のみ ) を算出

( 2 ) 鍵実体のMONEYフィールドから、決算額の減算処理を行なう。

## 【 0 1 5 2 】

## 9. 本発明チップ API仕様 (データ型定義)

B	符号付き1バイト整数
UB	符号なし1バイト整数
H	符号付き2バイト整数
UH	符号なし2バイト整数

本発明チップID	16バイト整数	10
SID	符号なし1バイト整数 (Session Id)	
FID	符号なし2バイト整数 (File/Folder Id)	
RID	符号なし2バイト整数 (Record Id)	
CACL	符号なし2バイト整数 (Chip Access Control List)	
FACL	符号なし2バイト整数 (File Access Control List)	
KEYACL	符号なし2バイト整数 (Key Object Access Control List)	20
TIME	符号なし4バイト整数 (2000年1月1日AM0:00からの秒数の積算)	
ERR	符号つき2バイト整数	

## ●C言語表現による定義例

```
#typedef unsigned char UB;
#typedef unsigned short UH;
#typedef unsigned long UW;

#typedef struct etronid{
    UB item[16];
}ETRONID; /* 本発明チップID */

#typedef UB SID; /* セッションID */
```

```

#typedef UB FID;          /* ファイルID */

#typedef UH RID;         /* レコードID */

#typedef UH CACL;        /* チップアクセス制御リスト */
#typedef UH FACL;        /* ファイルアクセス制御リスト */
#typedef UH KEYACL;      /* 鍵実体アクセス制御リスト */

#define ISSUER_MODE 1    /* 発行者モード */
#define OWNER_MODE 2    /* 所有者モード */

#define FILE_DYNAMIC 0x01 /* 動的マルチレコードファイル */
#define FILE_STATIC 0x02 /* 静的シングルレコードファイル */

#typedef UH TME;        /* 時刻 */

```

## 10. 本発明チップ API仕様 (命令定義)

### 10.1 セッション管理命令群

#### セッション構築

#### Open/Confirm Session

##### [ 機能概要 ]

本発明チップに対して安全なセッション通信路を構築する。1パスで済む認証を用いる場合は、eopn\_sesを、2パス必要な認証を用いる場合はeopn\_sesとecnf\_sesをこの順序で用いる。

##### 【 0 1 5 3 】

このセッション構築によって、Callerである本発明チップ搭載サービスクライアント側と、Callee側である本発明チップ側とでセッション中のみ有効な一時的な共通暗号鍵が共有される。

##### 【 0 1 5 4 】

認証の方式、暗号の種類は、パケット内のパラメータによって選択可能。本発明チップ側で利用できない認証 / 暗号が指定された場合はエラーとなる。

##### 【 0 1 5 5 】

## [関数表記]

```
ERR eopn_ses(ETRONID destId,ETRONID srcId,TIME t,
             UB authMode,UB authAlgorithm,UB sessionAlgorithm,
             UH len,UB*authData,
             UH*rlen,UB**rAuthData);
```

destId	命令の発行対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId の本発明チップ	本命令を呼び出す本発明チップ搭載サービスクライアント  (Source 本発明チップID)	
t	時刻	
authMode	認証モードを指定	20
authAlgorithm	認証に使うアルゴリズムの指定子	
sessionAlgorithm	認証後のセッションにおける暗号アルゴリズムの指定子	
len	行きのパケット長 (オクテット数)	
authData	認証用に本発明チップ搭載サービスクライアントから本発明チップに渡すデータ (len-16)	30
rlen	戻りパケット長 (オクテット長)	
rAuthData	認証用に本発明チップから本発明チップ搭載サービスクライアントに戻されたデータ (rlen-8)	

## [パラメーター値]

## □authMode

0x01	ISSUER	発行者モード	40
0x02	OWNER	所有者モード	

□authAlgorithm

- 0x01 予約
- 0x02 RSA (証明書を使ったオフライン認証)
- 0x03 RSA (CRLを確認するオンライン認証)
- 0x04 予約

10

□sessionAlgorithm

- 0x01 DES
- 0x02 3DES
- 0x03 予約
- 0x04 予約
- 0x05 予約

20

[返り値]

1パス認証の場合

- > 0 セッションID (正常終了時)
- < 0 エラーコード

2パス認証の場合

- <= 0 エラーコード

[送りパケット形式]

30

【表 2 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x01)	Command ID (0x01)		Len (オクテット)		予約	
t				auth Mode	auth Algo- rithm	ses Algo- rithm	auth Data
auth Deta							
.....							
auth Deta							

40

【表 2 9】

フィールド名	データ長	意味
Command ID	2(=0x01)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
authmode	1	認証用モード
authAlgorithm	1	認証アルゴリズム指定子
sessionAlgorithm	1	セッション暗号アルゴリズム指定子
authData	len-16	認証用データ

10

[ 戻りパケット形式 ]

【表 3 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x01)	Error Code		rLen (オクテット)		予約	
t				auth Mode	auth Algo- rithm	ses Algo- rithm	auth Data
rAuthData							
.....							
rAuthData							

20

30

【表 3 1】

フィールド名	データ長	意味
Error Code	2	エラーコード (、セッションID (1パスの場合))
rLen	2	戻りパケット長 (オクテット数)
rAuthData	rLen-8	認証用データ

40

[関数表記]

```
ERR ecnf_ses(ETRONID destId,ETRONID srcId,TIME t,
              UH len,UB*authData,
              UH*rLen,UB**rAuthData);
```

destId 命令の対象である本発明チップのチップID  
(Destination 本発明チップID) 10

srcId 本命令を呼び出す本発明チップ搭載サービスクライアントの本  
発明チップID  
(Source 本発明チップID)

t 時刻

len 行きのパケット長 (オクテット数)

authData 認証用に本発明チップ搭載サービスクライアントから本発明チ  
ップに渡すデータ (len-12) 20

rLen 戻りパケット長 (オクテット数)

rAuthData 認証用に本発明チップから本発明チップ搭載サービスライア  
ントに戻されたデータ (rLen-8)

[返り値]

> 0 セッションID (正常終了時)

< 0 ラーコード

[ 送りパケット形式 ]

【表 3 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x02)	Command ID (0x02)		len (オクテット)		予約	
t				authData			
.....							
authData							

【表 3 3】

フィールド名	データ長	意味
Command ID	2(=0x02)	コマンドデータ
len	2	パケット長 (オクテット)
t	4	時刻
authData	len-12	認証用データ

10

## [ 戻りパケット形式 ]

【表 3 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x02)	Error Code		rlen		予約	
rAuthData							
.....							
rAuthData							

20

【表 3 5】

フィールド名	データ長	意味
ErrorCode	2	エラーコード、セッション ID
rlen	2	戻りパケット長 (オクテット数)
rAuthData	len-8	認証用データ

30

## [ 機能詳細 ]

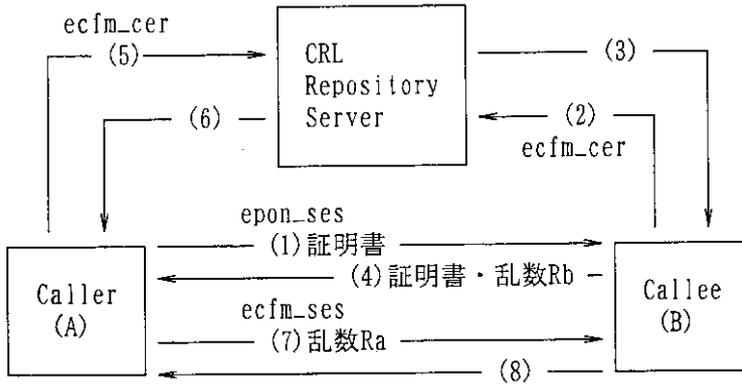
open/confirm session命令は、本発明チップ搭載サービスクライアントと本発明チップが相互認証した上で安全な暗号通信路としてのセッションを確立する。具体的な動作は、認証・暗号方式に依存し、その方式は多くの場合本発明Nチップハードウェアの暗号支援機能に依存する(実装依存)。セッション確立プロトコルは、発行者モードによる認証と、所有者モードによる認証があり、open session命令のauthmodeパラメータによって区別される。

40

## 【 0 1 5 6 】

公開鍵暗号による発行者モード(authmode=ISSUER)認証

【表 3 6】





## (4) eopn\_ses(A←B)

## 《パラメータ例》

- Error Code           E\_OK or not
- authMode            0x02 (ISSUER)
- authAlgorithm       0x03 (RSA:CRLを確認するオンライン認証)           10
- sessionAlgorithm    0x02 (3DES)
- authData            1. BのtTP公開鍵証明書  
                      2. 64ビットの乱数R bをAの公開鍵でRSA暗号化した

もの

- MACトレーラ        チェックサム

## 《A側の動作》

20

- RのeTP証明書の正当性の確認
- eTP証明書の期限確認
- CRLリストを確認→(5)へ
- CA局の公開鍵でeTP証明書の署名を確認
- 証明書からBの公開鍵を取り出す

- authData2. で受信した乱数R bをAの秘密鍵で復号化して取り出す           30
- 取り出した乱数Rbはセッション暗号の3DESの秘密鍵の一部にする。

【 0 1 5 7 】

- 取り出した乱数Rbのデジタル署名をAの秘密鍵で作成する。

【 0 1 5 8 】

## (5) ecfm\_cer (A→CRL Rep. Server)

## 《パラメータ例》

- ・ checkMode            0x00
- ・ certificateId        \*\*\*\*\*
- ・ MACトレーラ        チェックサム

10

## (6) ecfm\_cer (A←CRL Rep. Server)

## 《パラメータ例》

- ・ Error Code            E\_OK or not
- ・ MACトレーラ        チェックサム

## (7) ecfm\_ses (A→B)

20

## 《パラメータ例》

- ・ authData            1. 64ビットのR<sub>a</sub>をBの公開鍵でRSA暗号化したもの、  
と、  
2. Bから受け取った乱数の、Aの秘密鍵によるデジタル署名を、Bの公開鍵でRSA暗号化したものを、アペンドして送る
- ・ MACトレーラ        チェックサム

30

## 《B側の動作》

- ・ authDataで受信したデータをAの公開鍵で復号化してR<sub>a</sub>とデジタル署名を取り出す
- ・ 取り出したR<sub>a</sub>はセッション暗号の3DESの秘密鍵の一部にする。  
【 0 1 5 9 】
- ・ 取り出したR<sub>a</sub>のデジタル署名をBの秘密鍵で作成する。  
【 0 1 6 0 】
- ・ 取り出した署名をAの公開鍵で確認する。  
【 0 1 6 1 】

## (8) ecfm\_ses (Caller←Callee)

## 《パラメータ例》

- ・ Error Code                    E\_OK or not
- ・ authData                    Aから受け取った乱数Raの、Bの秘密鍵によるデジタル署名
- ・ MACトレーラ                チェックサム

10

## 《A側の動作》

- ・ 受け取った署名をBの公開鍵で確認する。

【 0 1 6 2 】

## ■（補足）認証アルゴリズムの詳細

### ■アルゴリズム詳細

- ・暗号・認証の鍵

秘密鍵  $s k = d$

公開鍵  $p k = (e, n)$

10

- ・平文  $m$  に対する暗号アルゴリズム

$E_{p k}(m) = m^e \bmod n$

- ・暗号文  $c$  に対する復号アルゴリズム

$D_{s k}(C) = c^d \bmod n$

- ・メッセージ  $M$  に対する署名生成アルゴリズム

20

$f_{s k}(M) = (h(M))^d \bmod n$

※ 但し、ここで  $h(M)$  は MD5 や SHA-1 などのハッシュ関数

- ・メッセージ  $M$  に対する署名  $s$  の検証アルゴリズム

30

$g_{p k}(M, s) = \text{if}(h(M) == s^e \bmod n) \text{ then } 1 \text{ else } 0 \text{ endif}$

### ■各ノードが所有する情報一覧

- ・CA局 (C)

CAの公開鍵  $p k_c$  (130B)

CAの秘密鍵  $s k_c$  (128B)

40

- ・Caller側 (A)

Aの公開鍵	$pk_a$	(130B)
Aの秘密鍵	$sk_a$	(128B)
Aの証明書	$cer_a$	(304B)
Aの証明書におけるCA局の署名	$s_{(a,c)}$	(128B)
CA局の公開鍵	$pk_c$	(130B)
相互認証中に発生する乱数 $R_a$		(8B)

10

• Callee側 (B)

Bの公開鍵	$pk_b$	(130B)
Bの秘密鍵	$sk_b$	(128B)
Bの証明書	$cer_b$	(304B)
Bの証明書におけるCA局の署名	$s_{(b,c)}$	(128B)
CA局の公開鍵	$pk_c$	(130B)
相互認証中に発生する乱数 $R_b$		(8B)

20

■ eTPによるRSAベースの認証アルゴリズム詳細

eopn\_ses: (A) → (B)

[送信情報]	$(cer_a)$ ※ 計304バイト
Aの証明書	$cer_a$ : including $PK_a$ and $s_{(a,c)}$

30

[Bの動作]

証明書の期限確認	
証明書の署名確認	$g_{(pk_c)}(cer_a, s_{(a,c)})$
証明書のCRLを確認	

eopn\_ses: (A) ← (B)

40

[送信情報]	$(cer_b   rb)$ ※ 計432バイト
--------	--------------------------

Bの証明書	$cer_b:$	
	including $pk_b$ and $s(b, c)$	
Bで生成した乱数( $R_b$ )	$rb = E_{(pk_a)}(R_b)$	
[Aの動作]		
証明書の期限確認		
証明書の署名確認	$g_{(pk_c)}(cer_b, s(b, c))$	10
証明書のCRLを確認		
メッセージ( $rb$ )の復号	$Rb' = D_{(sk_a)}(rb)$	
受け取った乱数( $Rb'$ )の署名作成	$srb = f_{(sk_a)}(Rb')$	
$ecfm\_ses: (A) \rightarrow (B)$		
[送信情報]	$(x_{a1}   x_{a2})$ ※ 計256バイト	20
Aで生成した乱数( $Ra$ )と署名( $srb$ )	$x_{a1} = E_{(pk_b)}(Ra)$ $x_{a2} = E_{(pk_b)}(srb)$	
[Bの動作]		
メッセージ( $x_a$ )の復号	$Ra' = D_{(sk_b)}(x_{a1})$ $srb' = D_{(sk_b)}(x_{a2})$	
受け取った署名( $srb$ )の検証	$g_{(pk_a)}(Rb, srb')$	30
受け取った乱数( $Ra'$ )の署名作成	$sra = f_{(sk_b)}(Ra')$	
$ecfm\_ses: (A) \leftarrow (B)$		
[送信情報]	$(x_b)$ ※ 計128バイト	
Bで生成した署名( $sra$ )	$x_b = E_{(pk_a)}(sra)$	
[Aの動作]		
メッセージ( $x_b$ )の復号	$sra' = D_{(sk_a)}(x_b)$	40
受け取った署名( $sra$ )の検証	$g_{(pk_b)}(Ra, sra')$	

これで、A、B双方に( $Ra, Rb$ )のペアが残され、ここから、一定のアルゴリズムでセッション共通鍵を生成する。

【0163】

セッション確立後の暗号通信パラメータの設定

(例) 3DESの場合

Ra-Rb-RaによるE-D-E方式による3DES暗号とする。この時セッションIDはeopn\_ses( )の往パケットのsid欄に格納されていた値を利用する。つまり最初の呼び出したcallerが指定したsession idをそのまま利用することになる。

【0164】

(例) DESの場合

Rbを鍵としたDES暗号とする。

【0165】

この時セッションIDはRaを利用する。

【0166】

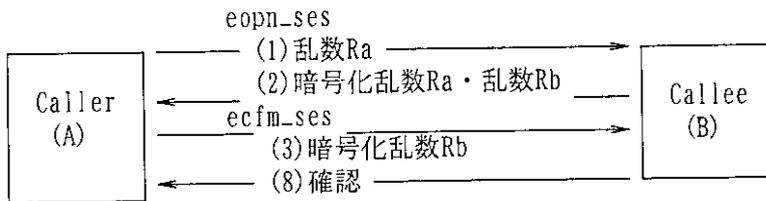
秘密鍵暗号による所有者モード(authmode=OWNER)認証

10

eopn\_ses、ecfm\_sesを用いた2パス式の所有者認証によるセッションの確立動作の例を以下に示す。ここでの方法は、本発明チップが所有者に依存したパスワードを保持し、それを利用して認証を行う。認証が開始された時に、本発明チップが乱数を発生してR/Wに送信する。その後、R/W側でその乱数を用いてパスワードを暗号化して、本発明チップに送る方式を基本とする。

【0167】

【表37】



20

(1) eopn\_ses(A B)

乱数 Ra を生成し、

- ・ ID\_A
- ・ Ra

を B に送る。

30

【0168】

(2) eopn\_ses(A B)

乱数 Rb を生成し、

- ・ X1=E1\_Key(ID\_A | Ra)
- ・ X2=E2\_P(Pb)

を計算し、A に戻す

(3) ecfm\_ses(A B)

ア . D1\_Key(X1)で復号して、Ra を確認する

(正しい本発明チップであることを認証)。

【0169】

40

イ . R\_B=D2\_P(X2)を計算しX2を復号してRbを取り出す。

【0170】

ウ . R\_Bを部分ビット反転などで変形したR'bを生成し、

- ・ Y=E3\_P(R'b)

を生成しBに送る。

【0171】

(4) cefm\_ses(A B)

D3\_p(Y)で復号して、R'bを確認(正しい所有者が本発明チップ搭載サービスクライアントを経由して操作していることを認証)。

【0172】

50

R a からセッションID(sid)を生成し、Aに戻す

ここで、"|"は、ビット列をappendしてつなげるという意味。例えば、  
A="010010011"、B="00100101"であれば、  
(A | B)="01001001100100101"である。

【0173】

ID\_A, ID\_Bはそれぞれ、本発明チップ搭載サービスクライアント(A)、本発明チップ

(B)の本発明チップID(128ビット)である。

【0174】

E2をDESだとするならば、E3は引数をビット反転してDESする。

10

【0175】

セッション終了

Close Session

[機能概要]

セッションを終了する。

【0176】

[関数表記]

```
ERR ecl_ses(SID sid, ETRONID destId, ETRONID srcID, TIME t,
            UH len, UH*rlen);
```

20

<b>destId</b>	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
<b>srcId</b>	本命令を呼び出す本発明チップ搭載サービスクライアント の本発明チップID (Source 本発明チップID)	
<b>t</b>	時刻	30
<b>len</b>	行きのパケット長 (オクテット数)	
<b>rlen</b>	返りのパケット長 (オクテット数)	

[返り値]

(= 0 エラーコード

40

[送りパケット形式]

【表38】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x03)		Len (12)		予約	
t							

【表 3 9】

10

フィールド名	データ長	意味
Command ID	2 (=0x03)	コマンドコード
len	2 (=12)	パケット長 (オクテット)
t	4	時刻

20

[ 戻りパケット形式 ]

【表 4 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		Len (8)		予約	

【表 4 1】

30

フィールド名	データ長	意味
ErrorCode	2	エラーコード、セッションID
rLen	2	戻りパケット長 (オクテット数)

[ アクセス制御 ]

この命令は、セッションを構築した本発明チップ搭載クライアントであれば、常に利用できる。

40

【 0 1 7 7 】

[ 機能詳細 ]

確立しているセッションを終了する。

【 0 1 7 8 】

10.2 トランザクション管理命令群

トランザクションセッションの構築

Open/Confirm Transaction

[ 機能概要 ]

本発明チップに対してトランザクションセッションを構築する。1パスで済む認証を用い

50

る場合は、eopn\_traを、2パス必要な認証を用いる場合はeopn\_traとecnf\_traとをこの順序で用いる。

【0179】

基本的な動作は、eopn\_ses,ecfm\_sesと同様である。ただ、その後の処理のロールバックが可能になるようになってるところが異なる。

【0180】

[関数表記]

```
ERR eopn_tra(ETRONID destId,ETRONID srcId,TIME t,
             UB authMode,UB authAlgorithm,UB sessionAlgorithm,
             UH len,UB*authData,UH*rLen,UB**rAuthData);
```

destId	命令の発行対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップ	
	(Source 本発明チップID)	20
t	時刻	
authMode	認証モードを指定	
authAlgorithm	認証に使うアルゴリズムの指定子	
sessionAlgorithm	認証後のセッションにおける暗号アルゴリズムの指定子	
len	行きのパケット長 (オクテット数)	30
authData	認証用に本発明チップ搭載サービスクライアントから本 発明チップに渡すデータ (len-16)	
rLen	戻りパケット長 (オクテット数)	
rAuthData	認証用に本発明チップから本発明チップ搭載サービスク ライアントに戻されたデータ (rLen-8)	40

[パラメータ値]

eopn\_ses参照

[返り値]

1パス認証の場合

>0 セッションID (正常終了時)

<0 エラーコード

2パス認証の場合

<=0 エラーコード

[ 送りパケット形式 ]

【表 4 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x01)	Command ID (0x04)		len		予約	
t				auth Mode	auth Algo- rithm	ses Algo- rithm	予約
authData							
.....							
authData							

10

20

【表 4 3】

フィールド名	データ長	意味
Command ID	2(=0x04)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
authmode	1	認証用モード
authAlgorithm	1	認証アルゴリズム指定子
sessionAlgorithm	1	セッション暗号アルゴリズム指定子
authData	len-16	認証用データ

30

40

[ 戻りパケット形式 ]

【表 4 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x01)	Error Code		rlen (オクテット)		予約	
rAuthData							
.....							
rAuthData							

10

【表 4 5】

フィールド名	データ長	意味
Error Code	2	エラーコード (、セッションID (1パスの場合))
rlen	2	戻りパケット長 (オクテット数)
rAuthData	rlen-8	認証用データ

20

[関数表記]

```
ERR ecnf_tra(ETRONID destId,ETRONID srcId,TIME t,
             UH len,UB*authData,
             UH*rLen,UB**rAuthData);
```

destId            命令の対象である本発明チップのチップID  
                   (Destination 本発明チップID) 10

srcId            本命令を呼び出す本発明チップ搭載サービスクライアント  
 の本発明チップID  
                   (Source 本発明チップID)

t                時刻

len              行きのパケット長 (オクテット数)

authData        認証用に本発明チップ搭載サービスクライアントから本発  
 明チップに渡すデータ (len-12) 20

rLen             戻りパケット長 (オクテット数)

rAuthData       認証用に本発明チップから本発明チップ搭載サービスクラ  
 イアントに戻されたデータ (rLen-8)

[返り値]

> 0            セッションID (正常終了時)

< 0            エラーコード

[ 送りパケット形式 ]

【表 4 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x02)	Command ID (0x05)		len (オクテット)		予約	
t				authData			
.....							
authData							

【表 4 7】

フィールド名	データ長	意味
Command ID	2(=0x02)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
authData	len-12	認証用データ

10

[ 戻りパケット形式 ]

【表 4 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial (0x02)	Error Code		rlen (オクテット)		予約	
rAuthData							
.....							
rAuthData							

20

【表 4 9】

フィールド名	データ長	意味
Error Code	2	エラーコード (、セッションID)
rlen	2	戻りパケット長 (オクテット数)
rAuthData	rlen-8	認証用データ

30

[ 機能詳細 ]

認証、暗号化に関しては、eopn\_ses、ecfm\_sesと同様。

【 0 1 8 1 】

但し、トランザクションが確立した後は、以下の本発明チップ APIだけが発行可能で、その他のものを発行すると、エラーになる。

40

【 0 1 8 2 】

- ・ Create File        ecre\_fil
- ・ Delete File        edel\_fil
- ・ Create Record     ecre\_rec
- ・ Delete Record     edel\_rec
- ・ Update Record     eupd\_rec

ここで発行された命令列は、ecom\_traが発行された時に反映され、eabo\_traが発行された時、または一定時間待ってecom\_traが発行されずタイムアウトしたときには完全にロールバックすることが保証されなければならない。

【 0 1 8 3 】

50

トランザクション終了

Commit/Abort Transaction

[ 機能概要 ]

トランザクションを終了する。コミットして終了する場合、ecom\_tra、アボートして終了する場合は、cabo\_tarを使う。

【 0 1 8 4 】

[関数表記]

ERR ecom\_tra(SID sid,ETRONID destId,ETRONID srcId,TIME t,  
UH len,UH\*rlen);

10

destId 命令の対象である本発明チップのチップID  
(Destination 本発明チップID)

srcId 本命令を呼び出す本発明チップ搭載サービスクライアントの  
本発明チップID  
(Source 本発明チップID)

t 時刻

20

len 行きのパケット長 (オクテット数)

rlen 返りのパケット長 (オクテット数)

[返り値]

<= 0 エラーコード

30

[ 送りパケット形式 ]

【表 5 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
予約	Serial No.	Command ID (0x06)		len (12)		予約	
t							

40

【表 5 1】

フィールド名	データ長	意味
Command ID	2 (=0x06)	コマンドコード
len	2 (0x0C)	パケット長 (オクテット数)
t	4	時刻

10

[ 戻りパケット形式 ]

【表 5 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

【表 5 3】

20

フィールド名	データ長	意味
Error Code	2	エラーコード (、セッションID)
rlen	2 (0x08)	戻りパケット長 (オクテット数)

[関数表記]

ERR eabo\_tra(SID sid,ETRONID destId,ETRONID srcID,TIME t,  
UH len,UH\*rlen);

destId 命令の対象である本発明チップのチップID  
(Destination 本発明チップID)

srcId 本命令を呼び出す本発明チップ搭載サービスクライアントの  
本発明チップID 10  
(Source 本発明チップID)

t 時刻

len 行きのパケット長 (オクテット数)

rlen 返りのパケット長 (オクテット数) 20

[返り値]

<= 0 エラーコード

[ 送りパケット形式 ]

【表 5 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x07)		len (12)		予約	
t							

30

【表 5 5】

フィールド名	データ長	意味
Command ID	2 (=0x07)	コマンドコード
len	2 (0x0C)	パケット長 (オクテット数)
t	4	時刻

40

[ 戻りパケット形式 ]

【表 5 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code	rlen (8)		予約		

【表 5 7】

フィールド名	データ長	意味
Error Code	2	エラーコード (、セッションID)
rlen	2	戻りパケット長 (オクテット数)

10

## [ アクセス制御 ]

この命令は、トランザクションを構築した本発明チップ搭載サービスクライアントであれば、常に利用できる。

## 【 0 1 8 5 】

20

## [ 機能詳細 ]

- ・ 確立しているトランザクションをコミット / アポートする。

## 【 0 1 8 6 】

- ・ コミットされた場合は、ecfm\_tra ~ ecom\_traの間で発行された命令をすべて、本発明チップの不揮発性記憶に反映させる。

## 【 0 1 8 7 】

- ・ アポートされた場合、または一定時間待ってもコミットが来ない時は、\_tra ~ ecom\_traの間で発行された命令をすべて、本発明チップの不揮発性記憶に反映させる。

## 【 0 1 8 8 】

## 10.3 ファイル管理命令群

30

## [ 実装制限事項 ]

本システムにおけるファイルは、以下の制限を持つ。

## 【 0 1 8 9 】

- ・ 使えるファイルの数の上限      50
- ・ ファイルID                      0 ~ 49
- ・ ファイルのサイズ上限            256 ( オクテット )

## ファイルの作成

## Create File

## [ 機能概要 ]

( 空 ) ファイルを作成する。

40

## 【 0 1 9 0 】

## [関数表記]

```
ERR ecre_fil(SID sid,ETRONID destId,ETRONID srcId,TME t,
             UH len,FID fid,UH blk,UH cnt,FACL facl,UH*rlen);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
fid	作成するファイルID	20
fac1	ファイルアクセス制御リストの初期値	
blk	ファイルのスタートアドレス (今回は"1"固定)	
cnt	ファイル長 (オクテット数)	
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	30

## [返り値]

> 0 作成されたファイルID (正常終了)

< 0 エラーコード

[ 送りパケット形式 ]

【表 5 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x21)		len (24)		予約	
t				fid		予約	
fac1		予約		blk		cnt	

【表 5 9】

フィールド名	データ長	意味
Command ID	2(=0x21)	コマンドコード
len	2(=0x18)	パケット長 (オクテット数)
t	4	時刻
fid	2	ファイルID
facl	2	初期アクセス制御リスト
blk	2	0x0000
cnt	ファイルID	ファイル長 (オクテット)

10

## [ 戻りパケット形式 ]

【表 6 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

20

【表 6 1】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rlen	2(=0x08)	戻りパケット長 (オクテット数)

30

## [ アクセス制御 ]

・発行者モード (親フォルダの ISSUER 本発明チップID=Sessionを構築した S C の本発明チップID)であればいつでも利用できる。

## 【 0 1 9 1 】

・親フォルダのアクセス制御リストで許可されていれば、所有者モード・他者モードで利用できる。

40

## 【 0 1 9 2 】

## [ 動作の詳細 ]

fidで指定したファイルIDを持ったファイルを作成する。指定したfidが既に利用されている場合はエラーとなる。

## 【 0 1 9 3 】

ファイルの削除

Delete File

## [ 機能概要 ]

ファイルを削除する。

## 【 0 1 9 4 】

50

## [関数表記]

```
ERR edel_fil(SID sid,ETRONID destId,ETRONID srcID,TIME t,
             UH len,FID fid,UH*rlen);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
fid	削除するファイルID	20
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

[ 送りのパケット形式 ]

【表 6 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x22)		len (16)		予約	
t				fid		予約	

【表 6 3】

30

40

フィールド名	データ長	意味
Command ID	2(=0x22)	コマンドコード
len	2(=0x10)	パケット長 (オクテット数)
t	4	時刻
fid	2	削除するファイルID

10

## [ 戻りパケット形式 ]

【表 6 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

20

【表 6 5】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rlen	2(=0x08)	戻りパケット長 (オクテット数)

## [ アクセス制御 ]

30

・ edel\_fil が実行できるためには、以下の条件を満たす必要がある。二種類のアクセス制御リストが関連するので注意が必要である。

## 【 0 1 9 5 】

( C A S E 1 ) 所有者認証アクセスの場合

1. ルートフォルダが「所有者アクセス」で edel\_fil できるように許可されてなければならない。つまりルートフォルダの ACL2==1 でなければならない。

## 【 0 1 9 6 】

2. 対象ファイルが「所有者アクセス」で edel\_fil できるように許可されてなければならない。つまりそのファイルの ACL8==1 でなければならない。

## 【 0 1 9 7 】

( C A S E 2 ) 発行者認証アクセスの場合

1. ルートフォルダが「その他アクセス」で edel\_fil できるように許可されてなければならない。つまりルートフォルダの ACL3==1 でなければならない。

## 【 0 1 9 8 】

2. 対象ファイル側の ACL には関係しない。(常に OK)

( C A S E 3 ) その他アクセスの場合

1. ルートフォルダが「その他アクセス」で edel\_fil できるように許可されてなければならない。つまりルートフォルダの ACL3==1 でなければならない。

## 【 0 1 9 9 】

2. 対象ファイルが「その他アクセス」で edel\_fil できるように許可されてなければなら

50

ない。つまりそのファイルのACL9== 1 でなければならない。

【 0 2 0 0 】

[ 動作の詳細 ]

fidで指定したファイルIDを持ったファイルを削除する。ファイルの物理的位置を指す制御ブロックをクリアし、不揮発性記憶領域を解放するだけでなく、中を必ず0クリアまたは1クリアする。

【 0 2 0 1 】

ファイルの譲渡要求

Request File Transfer

[ 機能概要 ]

ファイルの内容を他の本発明チップ搭載コンテンツホルダに譲渡する。

【 0 2 0 2 】

[関数表記]

```
ERR atra_fil(SID sid, ETRONID destId, ETRONID srcID, TIME t,
              UH len, FID fid, ETRONID targetId, FID targetFid,
              UH*rlen);
```

sid	セッションID	20
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID	
	(Source 本発明チップID)	
t	時刻	30
fid	作成するファイルID	
targetId	ファイルの譲渡先の本発明チップID	
targetFid	譲渡するファイルの譲渡先でのファイルID	
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	40

[返り値]

> 0 実際に更新したデータ長 (正常終了)

< 0 エラーコード

[ パケット形式 ]

【表 6 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x23)		len (36)		予約	
t				fid		予約	
targetId							
targetId							
target Fid		予約					

10

【表 6 7】

フィールド名	データ長	意味
Command ID	2(=0x23)	コマンドコード
len	2(=0x24)	パケット長 (オクテット数)
t	4	時刻
fid	2	ファイルID
targetId	16	ファイルの譲渡先の本発明チップ ID
targetFildId	2	譲渡先のファイルID

20

[ 戻りパケット形式 ]

【表 6 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

30

【表 6 9】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rlen	2(=0x08)	戻りパケット長 (オクテット数)

40

[ 動作詳細 ]

このシステムコールが呼ばれたチップは、以下の一連の処理を行う。

【 0 2 0 3 】

譲渡相手のチップに対して、以下の本発明チップ API列を発行する。

50

## 【 0 2 0 4 】

- ・トランザクションセッションをはる。 eopn\_tra
- ecfm\_tra
- ・ファイルを作成する。 ecre\_fil
- ・レコードを作成する。 ecre\_rec
- ：
- ：
- ・トランザクションをコミットする ecom\_tra

10

途中で何らかの異常が検知された場合は、トランザクションはアボートされる。また、正常にコミット命令が到達しない限りセッション中の命令が反映されることもない。

## 【 0 2 0 5 】

ファイルのデータを暗号化・復号化する

Encode/Decode File

[ 機能概要 ]

本発明暗号化コンテンツが格納されているファイルを復号する。

## 【 0 2 0 6 】

## [関数表記]

```
ERR eenc_fil(SID sid,ETRONID destId,ETRONID srcId,TIME t,
             UB len,FID srcFid,FID destFid,FID keyId
             UB*rlen,UW*currentMoney,UW*payedMoney);
```

```
ERR ednc_fil(SID sid,ETRONID destId,ETRONID srcId,TIME t,
             UB len,FID srcFid,FID destFid,FID keyId
             UB*rlen,UW*currentMoney,UW*payedMoney);
```

10

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID	20
t	(Source 本発明チップID) 時刻	
srcFid	暗号化コンテンツが格納されているファイルID	
destFid	復号されたコンテンツを格納するファイルID	
keyId	復号処理に使う鍵実体が格納されているファイルID	30
currentMoney	鍵実体中に残されている金額残高	
payedMoney	今回決裁した金額	
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

## [返り値]

40

<= 0 エラーコード

[パケット形式]  
【表 70】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x34, 35)		len (84+署名長)		予約	
t				srcFid		予約	
destFid		予約		keyId		予約	

10

【表 7 1】

フィールド名	データ長	意味
Command ID	2 (0x34, 35)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
srcFid	1	元ファイルID
destFid1	1	出力先ファイルID
keyId	1	鍵実体 ID

20

[ 戻りパケット形式 ]

【表 7 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (16)		予約	
currentMoney				payedMoney			

30

【表 7 3】

40

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
r len	2	戻りパケット長 (オクテット数)
currentMoney	4	鍵実体中の金額残高
payedMoney	4	今回の決裁額

10

[ 機能詳細 ]

edec\_fil ( ) の動作

【表 7 4】

・暗号化コンテンツ(srcFid)

(A)コンテンツヘッダ
(B)課金ヘッダ
暗号化KEY(1)
暗号化KEY(2)
暗号化コンテンツ本体(1)
.....
暗号化コンテンツ本体(m)
(D)署名トレーラ



・復号化コンテンツ(destFid)

(A)コンテンツヘッダ
(B)課金ヘッダ
00000000000000000000000000000000
00000000000000000000000000000000
復号化コンテンツ本体(1)
.....
復号化コンテンツ本体(m)
(D)署名トレーラ

20

30

eenc\_fil ( ) の動作

【表 7 5】

・元コンテンツ(srcFid)

(A)コンテンツヘッダ
(B)課金ヘッダ
元のKEY(1)
元のKEY(2)
「生」コンテンツ本体(1)
.....
「生」コンテンツ本体(m)



・暗号化コンテンツ(destFid)

(A)コンテンツヘッダ
(B)課金ヘッダ
暗号化KEY(1)
暗号化KEY(2)
暗号化コンテンツ本体(1)
.....
暗号化コンテンツ本体(m)
(D)署名トレーラ

10

コンテンツの暗号化は、「元のKEY」で行なう。

20

【0207】

KEYの暗号化は、指定した鍵実体の鍵で行なう。

【0208】

署名は、暗号化が終了したデータに関して作用する。

【0209】

HMAC with MD5の時の64ビット秘密鍵Kは、元のKEYを4回繰り返したものを使う。

【0210】

ファイルのアクセス制御リストの変更

Update File Mode

30

[機能概要]

ファイルのアクセス制御リストを変更する。

【0211】



フィールド名	データ長	意味
Command ID	2(=0x27)	コマンドコード
len	2(=0x14)	パケット長 (オクテット数)
t	4	時刻
fid	2	ファイルID
filacl	2	ファイルアクセス制御リスト

10

[ 戻りパケット形式 ]

【表 7 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code	rlen (8)		予約		

20

【表 7 9】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rlen	2(=0x08)	戻りパケット長 (オクテット数)

30

[ アクセス制御 ]

・対象ファイルに対して発行者モード (ファイルのISSUER 本発明チップID = sessionを構築したSCの本発明チップID) であればいつでも利用できる。

【0 2 1 2】

・対象ファイルのアクセス制御リストで許可されていれば、所有者モード・他者モードで利用できる。

【0 2 1 3】

[ 機能概要 ]

ファイルのアクセス制御リストを、指定した値に変更する。

【0 2 1 4】

定義済ファイルのリストを得る

List File ID

[ 機能概要 ]

定義済ファイルのリストを得る

40

## [関数表記]

```
ERR elst_fid(SID sid,ETRONID destId,ETRONID srcId,TIME t,
            UH len,FID fid,UH*rLen,UB**fileCtrlBlk);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
fid	対象のファイルID	20
fileCtrlBlk	対象のファイルの制御ブロック	
len	行きのパケット長 (オクテット数)	
rLen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

30

[ 送りパケット形式 ]

【表 8 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x2F)		len (16)		予約	
t				fid		予約	

40

【表 8 1】

フィールド名	データ長	意味
Command ID	2(=0x2F)	コマンドコード
len	2(16)	パケット長 (オクテット数)
t	4	時刻
fid	2	ファイルID

10

## [ 戻りパケット形式 ]

【表 8 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		ren		予約	
fileCtrlBlk							
.....							
fileCtrlBlk							

20

【表 8 3】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rln	2	戻りパケット長 (オクテット数)
fileCtrlBlk	*rln-8	対象のファイルの制御ブロック

30

## [ アクセス制御 ]

・対象ファイルに対して発行者モード (ファイルのISSUER 本発明チップID = Sessionを構築したSCの本発明チップID) であればいつでも利用できる。

40

## 【0215】

・対象ファイルのアクセス制御リストで許可されていれば、所有者モード・他者モードで利用できる。

## 【0216】

## 10.4 レコード管理命令群

レコードのデータを更新

Update Record

## [ 機能概要 ]

レコードの内容を変更する。

## 【0217】

50

## [関数表記]

```
ERR eupd_rec(SID sid, ETRONID destId, ETRONID srcId, TIME t,
             UH len, FID fid, UH blk, UH cnt, UB*data, UH*rLen);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
fid	変更するファイルID	20
blk	変更するレコード番号 (今回は= 1 固定)	
cnt	変更するレコードデータ長 (オクテット数)	
data	変更するデータの内容 (cntオクテット)	
len	行きのパケット長 (オクテット数)	
rLen	返りのパケット長 (オクテット数)	30

## [返り値]

> 0 実際に更新したデータ長 (正常終了)

< 0 エラーコード

[パケット形式]

【表 8 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x33)		len		予約	
t				fid		予約	
blk		cnt		data			
data							
.....							
data							

10

【表 8 5】

フィールド名	データ長	意味
Command ID	2(0x33)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
fid	2	ファイルID
blk	2	変更レコードID (今回は=1に固定)
cnt	2	データ長 (オクテット数)
data	cnt	書き込むデータ

20

30

【戻りパケット形式】

【表 8 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

40

【表 8 7】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rlen	2	戻りパケット長 (オクテット数)

50

## [ アクセス制御 ]

・対象レコードに対して発行者モード（レコードのISSUER 本発明チップID = Sessionを構築したSCの本発明チップID）であればいつでも利用できる。

## 【 0 2 1 8 】

・対象レコードのアクセス制御リストで許可されていれば、所有者モード・他で利用できる。

## 【 0 2 1 9 】

## [ 動作の詳細 ]

blkレコードで、始めからcntオクテット分のデータを書き込む要求を出す。cntは、ハードウェアの条件によって上限が設けられることがある。実際に書き込まれたオクテット数がrCntに返る。

## 【 0 2 2 0 】

レコードからデータを読む

Read Record

## [ 機能概要 ]

レコードの内容を読み出す。

## 【 0 2 2 1 】

[関数表記]

```
ERR area_rec(SID sid,ETRONID destId,ETRONID srcId,TIME t,
              UH len,FID fid,UH blk,UH cnt,
              UH*rLen,UH*rCnt,UB*rData);
```

sid	セッションID	10
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID	
	(Source 本発明チップID)	
t	時刻	20
fid	読み出すファイルID	
blk	読み出すレコードレID	
cnt	読み出すデータの長さ (オクテット数)	
rCnt	実際に読み出したデータの長さ (オクテット数)	
rData	読み出したデータ (rCntオクテット)	30
len	行きのパケット長 (オクテット数)	
rLen	返りのパケット長 (オクテット数)	

[返り値]

<= 0 エラーコード

[ パケット形式 ]

【表 8 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x33)		len		予約	
t				fid		予約	
blk		cnt					

【表 8 9】

フィールド名	データ長	意味
Command ID	2(0x34)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
fid	2	ファイルID
blk	2	レコードID
cnt	2	データ長 (オクテット)

10

[ 戻りパケット形式 ]

【表 9 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	
rCnt		予約		rData			
rData							
.....							
rData							

20

30

【表 9 1】

フィールド名	データ長	意味
Error Code	2	読み出したデータ長 (オクテット数)、エラーコード
rCnt	2	実際に読み出したデータ長 (オクテット数)
rData	rCnt	読み出したデータ

40

[ アクセス制御 ]

・対象レコードに対して発行者モード (レコードのISSUER 本発明チップID =Sessionを構築したSCの本発明チップID) であればいつでも利用できる。

【 0 2 2 2 】

・対象レコードのアクセス制御リストで許可されていれば、所有者モード・他者モードで利用できる。

50

## 【 0 2 2 3 】

## [ 動作の詳細 ]

blkレコードの最初から、cntオクテット分のデータを読み込む要求を出す。cntは、ハードウェアの条件によって上限が設けられることがある。実際に読み込まれたオクテット数がrCntに返る。

## 【 0 2 2 4 】

通常は、セッションが確立したところで発行されて実行されるが、タッチ&ゴー応用の特化し、セッションを確立していない本発明チップ搭載サービスクライアントでも、sid=0で本命令を呼び出すことができる。この場合、レコードのアクセス制御リストで、その他権限での読み出しが許可されていると、命令の実行は成功する。

10

## 【 0 2 2 5 】

## 10.5 鍵実体管理命令群

## [ 実装制限事項 ]

本システムにおける鍵実体は、以下の制限を持つ。

## 【 0 2 2 6 】

- ・使えるファイル数の上限 10
- ・鍵実体ID 0~9
- ・鍵実体のサイズ上限 256 (オクテット)

## 鍵実体を作成する

### Create Key Object

10

#### [機能概要]

#### [関数表記]

```
ERR ecre_key(SID sid, ETRONID destId, ETRONID srcId, TIME t,
             UH len, UH kid, UH keyAlgorithm, UB*key,
             UH bankId, KEYACL acl, UL initMoney Val,
             UH*rlen);
```

20

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID	30
	(Source 本発明チップID)	
t	時刻	
kid	鍵実体ID	
KeyAlgorithm	暗号アルゴリズム指定子	
key	暗号鍵	
bankId	銀行ID	40
initMoney Val	初期金額	

**acl** 初期アクセス制御リスト値 (最初のバージョンでは未使用)

**len** 行きのパケット長 (オクテット数)

**rLen** 返りのパケット長 (オクテット数)

[ 返り値 ]

<= 0 エラーコード

10

[ パケット形式 ]

【表 9 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x61)		len		予約	
t				kid		keyAlgorithm	
KEY (1)							
KEY (2)							
bankId		acl		initMoneyVal			

20

【表 9 3】

フィールド名	データ長	意味
Command ID	2 (0x61)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
kid	2	鍵実体ID
keyAlgorithm	2	暗号アルゴリズム指定子
key	16	暗号鍵
bankId	2	銀行ID
acl	2	初期アクセス制御リスト値 (最初のバージョンでは未使用)
initMoneyVal	4	初期金額

30

40

[ 戻りパケット形式 ]

【表 9 4】

50

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

【表 9 5】

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2	戻りパケット長 (オクテット数)

10

鍵実体を削除する  
Delete Key Object  
[ 機能概要 ]

## [関数表記]

```
ERR edel_key(SID sid,ETRONID destId,ETRONID srcId,TIME t,
             UB len,UH kid,
             UH*bankId,UL*remainingMoney,UH*rsize,UB**bookList,
             UH*rlen);
```

sid	セッションID	10
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID	
	(Source 本発明チップID)	
t	時刻	20
kid	削除する鍵実体ID	
bankId	銀行ID	
remainingMoney	残金	
rsize	Distribution Listの長さ	
bookList	Distribution List	30
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

40

[パケット形式]

【表 9 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x62)		len		予約	
t				kid		予約	

【表 9 7】

10

フィールド名	データ長	意味
Command ID	2(0x62)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
kid	2	鍵実体ID

20

[ 戻りパケット形式 ]

【表 9 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (16)		予約	
BANK ID		予約		CurrentMoney 支払残高			

30

【表 9 9】

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2	戻りパケット長 (オクテット数)
bankId	2	銀行ID
currentMoney	4	現在の金額残高

40

鍵実体を更新する

Update Key Object

[ 機能概要 ]

指定した鍵実体の金額を更新する。

50

【 0 2 2 7 】

[関数表記]

```
ERR eupd_key(SID sid,ETRONID destId,ETRONID srcId,TIME t,
             UH len,UH kid,UL addMoney,
             UH*rlen,UL*currenMoney);
```

sid	セッションID	10
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	20
kid	更新する鍵実体ID	
addMoney	加算される金額	
currentMoney	加算後の金額	
len	行きのパケット長 (オクテット数)	30
rlen	返りのパケット長 (オクテット数)	

[返り値]

<= 0 エラーコード

[ パケット形式 ]

【表 1 0 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x63)		len		予約	
t				kid		予約	
addMoney							

40

【表 1 0 1】

50

フィールド名	データ長	意味
Command ID	2(0x63)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
kid	2	更新する鍵実体ID
addMoney	4	加算する金額

10

[ 戻りパケット形式 ]

【表 1 0 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (12)		予約	
currentMoney 加算後金額							

20

【表 1 0 3】

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2(12)	戻りパケット長 (オクテット数)
currentMoney	4	加算後金額

30

鍵実体の情報を読む

Read Key Object

[ 機能概要 ]

指定した鍵実体の金額情報を読み出す。

【 0 2 2 8 】

40

## [関数表記]

```
ERR area_key(SID sid,ETRONID destId,ETRONID srcId,TIME t,
              UH len,UH kid,UH*rlen,UL*currentMoney);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
kid	金額情報を読み出す対象である鍵実体ID	20
currentMoney	現在の金額	
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

[パケット形式]

【表 1 0 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x64)		len		予約	
t				kid		予約	

【表 1 0 5】

30

40

フィールド名	データ長	意味
Command ID	2(0x63)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
kid	2	読み出す鍵実体ID

10

[ 戻りパケット形式 ]

【表 1 0 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (12)		予約	
currentMoney							

20

【表 1 0 7】

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2(=12)	戻りパケット長 (オクテット数)
currentMoney	4	現在の金額

30

鍵実体のアクセス制御リストを更新する

Update Key Mode

[ 機能概要 ]

鍵実体のアクセス制御リストを更新する。

【 0 2 2 9 】

最初のバージョンでは未実装

40

## [関数表記]

```
ERR eupd_key(SID sid,ETRONID destId,ETRONID srcId,TIME t,
             UH len,UH kid,KEYACL keyacl,UH*rlen,);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
kid	アクセス制御リスト更新する対象である鍵実体ID	20
keyacl	更新する鍵実体アクセス制御リスト	
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

30

[パケット形式]

【表 1 0 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x65)		len		予約	
t				kid		予約	

40

【表 1 0 9】

フィールド名	データ長	意味
Command ID	2(0x65)	コマンドコード
len	2	パケット長 (オクテット数)
t	4	時刻
kid	2	更新する鍵実体ID
keyacl	2	鍵実体アクセス制御リスト

10

[ 戻りパケット形式 ]

【表 1 1 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	
currentMoney							

20

【表 1 1 1】

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2(=8)	戻りパケット長 (オクテット数)
currentMoney	4	加算後金額

30

定義済鍵実体のリストを得る

List Key Object ID

[ 機能概要 ]

定義済鍵実体のリストを得る

## [関数表記]

```
ERR elst_kid(SID sid, ETRONID destId, ETRONID srcId, TIME t,
            UH len, KID kid, UH*rLen, UB**keyCtrlBlk);
```

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
kid	対象の鍵実体ID	20
keyCtrlBlk	対象の鍵実体の制御ブロック	
len	行きのパケット長 (オクテット数)	
rLen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

30

[ 送りパケット形式 ]

【表 1 1 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x6F)		len (16)		予約	
t				kid		予約	

40

【表 1 1 3】

フィールド名	データ長	意味
Command ID	2(=0x6F)	コマンドコード
len	2(=16)	パケット長 (オクテット数)
t	4	時刻
kid	2	鍵実体ID

10

## [ 戻りパケット形式 ]

【表 1 1 4】

D 1	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen		予約	
keyCtrlBlk							
.....							
keyCtrlBlk							

20

【表 1 1 5】

フィールド名	データ長	意味
Error Code	2	ファイルID、エラーコード
rlen	2(=8)	戻りパケット長 (オクテット数)
keyrBlk	*rlen-8	対象のファイルの制御ブロック

30

## [ アクセス制御 ]

・対象鍵実体に対して発行者モード (ファイルのISSUER 本発明チップID =Sessionを構築したSCの本発明チップID) であればいつでも利用できる。

40

## 【 0 2 3 0 】

・対象鍵実体のアクセス制御リストで許可されていれば、所有者モード・他者モードで利用できる。

## 【 0 2 3 1 】

## 10.6 認証補助管理命令群

eTP証明書をチェックする

Confirm Certificate

## [関数表記]

```
ERR ecfm_cer(SID sid,ETRONID destId,ETRONID srcId,TIME t,
             UH len,UB checkMode,UH serial,UH caId,
             UB*rlen,UB*crl);
```

sid	セッションID	10
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID	
t	(Source 本発明チップID) 時刻	20
checkMode	破棄リストのチェックモード	
serial	確認する証明書の証明書番号	
caId	確認する証明書の発行者ID	
crl	破棄リスト (本発明チップ形式)	
len	行きのパケット長 (オクテット数)	30
rlen	返りのパケット長 (オクテット数)	

## [checkMode]

```
0x00   チェックだけを返す
0x01   対応する破棄リストを得る
```

## [返り値]

```
> 0   セッションID (正常終了時)
```

```
< 0   エラーコード
```

[パケット形式]

【表 1 1 6】

40

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x71)		len		予約	
t				checkMode		予約	
Serial		CA ID					

10

【表 1 1 7】

フィールド名	データ長	意味
Command ID	2(0x71)	コマンドコード
len	2	パケット長
t	4	時刻
checkMode	2	チェックモード
certificateId	16	チェックしたい証明書ID

20

[ 戻りパケット形式 ]

【表 1 1 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	
crl							
.....							
crl							

30

【表 1 1 9】

40

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2	破棄リスト長
crl	rlen-8	破棄リスト

50

次に記述するAPI命令は、その形式は本発明チップ API命令に準ずるが、必ずしも本発明チップ搭載サービスクライアントだけが発行する命令でなく、他の実体が発行する命令である。主に、ハードウェアに依存した管理、運用、また通信の下位階層をサポートするためのインタフェースが中心である。チップリーダーユニットが発行したり、製造時のコンピュータが発行するケースがある。

【 0 2 3 2 】

チップをポーリングする

Poll CHIP

[ 機能概要 ]

本発明チップの状態を読み出す（実装依存、運用依存）。

10

【 0 2 3 3 】

[関数表記]

ERR epol\_car(ETRONID\* destId,ETRONID srcId,TIME t);

destId            ポーリングしたときに返されるチップID

srcId             本命令を呼び出す本発明チップ搭載サービスクライアントの

本発明チップID

20

t                 時刻

[返り値 (例) ]

<= 0     エラーコード

[ パケット形式 ]

【表 1 2 0】

D 0	D 1	D 2	D 3
予約 (0x00)	Serial (0x00)	Com. ID (0x41)	予約

30

【表 1 2 1】

フィールド名	データ長	意味
Command ID	1(0x41)	コマンドコード

40

[ 戻りパケット形式 ]

【表 1 2 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID (0x00)	Serial (0x00)	Error Code	rLen (0x08)	cardVersion		予約	

【表 1 2 3】

フィールド名	データ長	意味
Error Code	1	エラーコード
cardVersion	2	カードバージョン
rLen	1	データパケット長(=0x08)

10

cardId、clientIdともにチップヘッダにつくので、本体の中には入らない。

## 【 0 2 3 4 】

20

## [ アクセス制御 ]

このAPIは、常に利用できる。

## 【 0 2 3 5 】

セッション中でなくてもよい。

## 【 0 2 3 6 】

## [ 機能の詳細 ]

epol\_carの戻りパケットのルーティングヘッダ部分のsrcId部分に、チップの本発明チップIDを格納して戻すため、誰でも本発明チップのIDを最初に取得するために用いることができる。

## 【 0 2 3 7 】

30

セッションの中でなく、誰でも発行できる。

## 【 0 2 3 8 】

チップを初期化する

Initialize CHIP

## [ 機能概要 ]

本発明チップを初期化するために、チップ管理部を作成する。

## 【 0 2 3 9 】

## [関数表記 (例) ]

```
ERR eini_car(ETRONID dstId,ETRONID srcId,TIME t,
             UB initKey1[8],UB initKey2[8],UB initPasswd[8],
             UB fileNum,UB keyObjLen,UB swapLen,CACL cac1,
             UB rsaSecretKey[128],UH len,UH*rlen);
```

destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	all”1”	
t	時刻	
initKey1	認証用共通鍵 (1)	
initKey2	認証用共通鍵 (2)	
initPasswd	初期パスワード	20
fileNum	チップ内に作成可能なファイル数の上限	
keyObjNum	鍵実体数の上限	
swapLen	不揮発性記憶の中のスワップ領域の大きさ	
cac1	チップのアクセス制御リスト	
rsaSecretKey	自カードの認証用(RSA)秘密鍵	
len	行きのパケット長 (オクテット数)	30
rlen	返りのパケット長 (オクテット数)	

## [返り値]

<= 0 エラーコード

[ パケット形式 ]

【表 1 2 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x51)		len (176)		予約	
t				予約			
認証用共通鍵(1)							
認証用共通鍵(2)							
パスワード							
cacl	file Num	keyObj Num	swap Len	予約	version		
自分用(RSA)秘密鍵(1)							
.....							
自分用(RSA)秘密鍵(16)							

10

20

【表 1 2 5】

フィールド名	データ長	意味
Command ID	2(0x51)	コマンドコード
len	2(176)	パケット長
t	4	時刻
initKey1	8	認証用共通鍵(1)
initKey2	8	認証用共通鍵(2)
initPasswd	8	認証用パスワード
fileNum	1	ファイル数上限(50)
keyObjNum	1	鍵実体数上限(10)
swapLen	1	スワップ領域
cacl	1	初期カードアクセス制御リスト
rsaSecretKey	128	自分用の認証用RSA鍵(1024bit)

30

40

[ 戻りパケット形式 ]

【表 1 2 6】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code	rlen (8)		予約		

【表 1 2 7】

フィールド名	データ長	意味
Error Code	2	エラーコード
rlen	2	返りパケット長

10

## [ 動作の詳細 ]

与えられた引数に応じて、システム制御ブロックを初期化する。

## 【 0 2 4 0 】

20

その他のメモリ領域は、すべて 0 クリアする。

## 【 0 2 4 1 】

## [ アクセス制御 ]

本発明チップのsource 本発明チップID=0xff...ff(all"1")の時のみ受け付ける。

## 【 0 2 4 2 】

自認証公開鍵証明書入換

Update My Certificate

## [ 機能概要 ]

本発明チップ自身の証明書を更新する。

## 【 0 2 4 3 】

30

[関数表記]

ERR eupd\_cer(ETRONID destId,ETRONID srcId,TIME t,  
UB\*certificate,UH len,UH\*rlen);

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
certificate	証明書	20
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

[返り値]

<= 0 エラーコード

[ パケット形式 ]

【表 1 2 8】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x52)		len (320)		予約	
t				予約			
証明書(1)							
.....							
証明書(38)							

【表 1 2 9】

フィールド名	データ長	意味
Command ID	2(0x52)	コマンドコード
len	2(304)	パケット長
t	4	時刻
certificate	304	証明書

10

[ 戻りパケット形式 ]

【表 1 3 0】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

20

[ 戻りパケット形式 (例) ]

【表 1 3 1】

フィールド名	データ長	意味
Error Code	2	エラーコード、セッションID
rlen	2	返りパケット長 (オクテット数)

30

C A局公開鍵入換

Update CA Public Kye

[ 機能概要 ]

C A局の公開鍵交換する。

【 0 2 4 4 】

[関数表記]

ERR eupd\_cpk(ETRONID destId,ETRONID srcId,TIME t,  
UB\*caPublicKey,UH len,UH\*rlen);

sid	セッションID	
destId	命令の対象である本発明チップのチップID (Destination 本発明チップID)	10
srcId	本命令を呼び出す本発明チップ搭載サービスクライアントの 本発明チップID (Source 本発明チップID)	
t	時刻	
caPublicKey	CA局の公開鍵	20
len	行きのパケット長 (オクテット数)	
rlen	返りのパケット長 (オクテット数)	

[返り値]

<= 0 エラーコード

[ パケット形式 ]

【表 1 3 2】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Command ID (0x53)		len (144)		予約	
t				予約			
CA局公開鍵 (1)							
.....							
CA局公開鍵 (16)							

【表 1 3 3】

フィールド名	データ長	意味
Command ID	2(0x53)	コマンドコード
len	2(144)	パケット長
t	4	時刻
certificate	128	RSA証明書(1024ビット)

10

[ 戻りパケット形式 ]

【表 1 3 4】

D 0	D 1	D 2	D 3	D 4	D 5	D 6	D 7
SID	Serial No.	Error Code		rlen (8)		予約	

20

[ 戻りパケット形式 (例) ]

【表 1 3 5】

フィールド名	データ長	意味
Error Code	2	エラーコード、セッションID
rlen	2	返りパケット長 (オクテット数)

30

暗号実装仕様

D E S

64ビットの秘密鍵から56ビットのDES鍵の生成方法

以下の通り、各オクテットの最下位ビットをパリティビットとする（しかしチップでは無視するだけでチェックはしない）。上位7ビットを8オクテット分足した56ビットをDESとして利用する。

【 0 2 4 5 】

・ KEY (1)

b7b6b5b4b3b2b1b0 b7b6b5b4b3b2b1 b0b7b6b5b4b3b2b1b0 b7b6b5b4b3b2b1b0  
 □□□□□□□■ □□□□□□□ □■□□□□□□□■ □□□□□□□■

・ KEY (2)

b7b6b5b4b3b2b1b0 b7b6b5b4b3b2b1 b0b7b6b5b4b3b2b1b0 b7b6b5b4b3b2b1b0  
 □□□□□□□■ □□□□□□□ ■□□□□□□□■ □□□□□□□■

10

■ RSA

RSA鍵のデータ形式

鍵について、以下のように表現する。

【 0 2 4 6 】

d = 1024ビット (128オクテット)

n = 1024ビット (128オクテット)

e = 0x0003 固定 (1オクテット)

秘密鍵sk=d (128オクテット)

公開鍵pk=e の長さ(1オクテット) | e (1オクテット) | n (128オクテット)

【 0 2 4 7 】

【発明の効果】

以上説明したように、本発明によれば、本発明の自律型 IC カードは、IC カード端末を介して接続される通信装置を自ら認識し、その通信装置と直接的に通信を行うので、正確な情報の安全な通信を保證することができる。また、本発明の自律型 IC カードが、通信装置と相互に認証処理を行うと共に通信装置との通信に係る情報に対して暗号化 / 復号化を施す暗号処理部を有する構成とした場合、更に、通信装置の正規性を判断できると共に、その自律型 IC カードと、その通信装置との間に物理的に介在する装置によるデータ内容の盗難やデータの改竄の可能性を十分に低減することができる。特に、そのデータが電子チケット等のいわゆる価値情報にかかるものである場合、その意義は大きい。また、暗号処理部が、認識された通信装置の種別に応じて、複数の認証処理及び複数の暗号処理の中から適した認証処理及び暗号処理を選択してそれらの処理を行う場合、認識された通信装置に適した処理を行うことができる。

30

【図面の簡単な説明】

【図 1】本発明における IC カードの一実施形態の論理的な構成を示す図である。

【図 2】本発明の自律型 IC カードによる IC カード間通信を行うシステムの構成を示す図である。

40

【図 3】本発明の自律型 IC カードによる通信を行う他のシステムの構成を示す図である。

【図 4】従来における IC カードと IC カード端末との信号のやり取りを説明するための図である。

【図 5】従来において、2 枚の IC カードが互いに通信を行う場合の信号のやり取りを示す図である。

【符号の説明】

1, 1 a, 1 b 自律型 IC カード

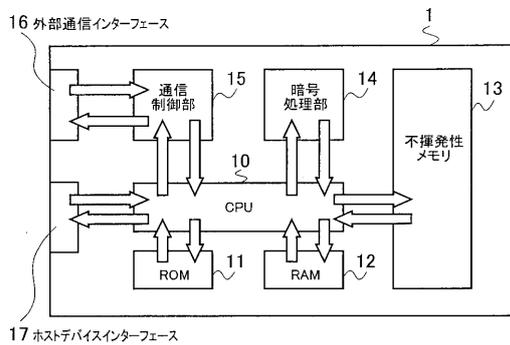
2, 2 a, 2 b IC カードリーダー / ライター

3 IC カード端末本体

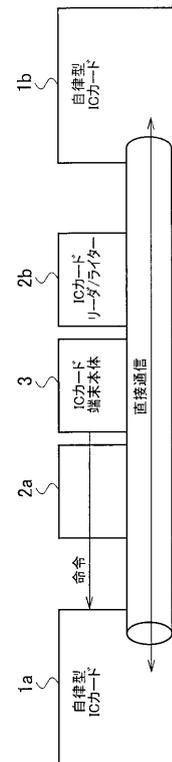
50

- 4 ネットワーク
- 5 通信装置
- 10 CPU
- 11 ROM
- 12 RAM
- 13 不揮発性メモリ
- 14 暗号処理部
- 15 通信制御部
- 16 外部通信インターフェース
- 17 ホストデバイスインターフェース
- 100 ICカード
- 100a ICカード
- 100b ICカード
- 200 ICカードリーダー/ライター
- 200a ICカードリーダー/ライター
- 200b ICカードリーダー/ライター
- 300 ICカード端末本体

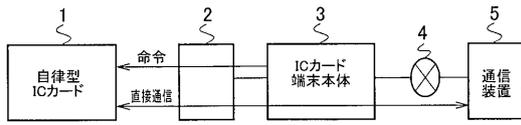
【図1】



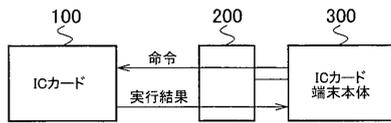
【図2】



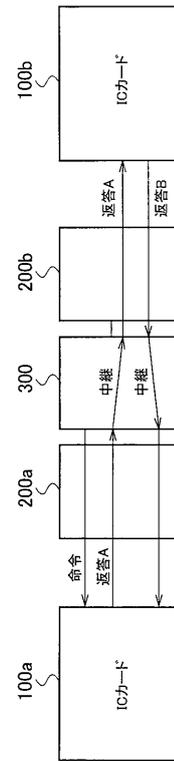
【図3】



【図4】



【図5】



---

フロントページの続き

- (72)発明者 石井 一彦  
東京都千代田区永田町二丁目1番1号 株式会社エヌ・ティ・ティ・ドコモ内
- (72)発明者 森 謙作  
東京都千代田区永田町二丁目1番1号 株式会社エヌ・ティ・ティ・ドコモ内
- (72)発明者 青野 博  
東京都千代田区永田町二丁目1番1号 株式会社エヌ・ティ・ティ・ドコモ内
- (72)発明者 本郷 節之  
東京都千代田区永田町二丁目1番1号 株式会社エヌ・ティ・ティ・ドコモ内

審査官 相崎 裕恒

- (56)参考文献 特開2001-167244(JP,A)  
特開2001-216459(JP,A)  
特表2002-517052(JP,A)  
特開2002-042061(JP,A)  
特開2002-063141(JP,A)  
特開2001-257673(JP,A)  
特開2001-243196(JP,A)  
特開2001-160828(JP,A)

- (58)調査した分野(Int.Cl., DB名)  
G06K 17/00-19/10