(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3801833号
(P3801833)
(45)発行日　平成18年7月26日(2006.7.26)　(24)登録日　平成18年5月12日(2006.5.12)

(51)Int.Cl.　F I
H04L 9/10 (2006.01)　H04L 9/00 621A
G09C 1/00 (2006.01)　G09C 1/00 620Z

請求項の数 5　(全 26 頁)

(21)出願番号　特願2000-35898(P2000-35898)
(22)出願日　平成12年2月14日(2000.2.14)
(65)公開番号　特開2001-230770(P2001-230770A)
(43)公開日　平成13年8月24日(2001.8.24)
審査請求日　平成14年12月11日(2002.12.11)

(73)特許権者　000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(74)代理人　100083806
弁理士　三好　秀和
(74)代理人　100100712
弁理士　岩▲崎▼　幸邦
(74)代理人　100100929
弁理士　川又　澄雄
(74)代理人　100108707
弁理士　中村　友之
(74)代理人　100095500
弁理士　伊藤　正和
(74)代理人　100101247
弁理士　高橋　俊一

最終頁に続く

(54)【発明の名称】マイクロプロセッサ

(57)【特許請求の範囲】

【請求項１】

外部メモリとのインタフェースを持つマイクロプロセッサであって、

マイクロプロセッサ内部に設けられた鍵テーブルである一時鍵バッファと、

当該マイクロプロセッサに埋め込まれた秘密鍵により、前記外部メモリ上に配置されたプログラム対応の復号鍵情報を復号して前記一時鍵バッファに読み込む第１の復号手段と、

前記外部メモリから読み込んだ暗号化された命令を、前記一時鍵バッファに格納された復号鍵によって復号してキャッシュメモリに格納する第２の復号手段と、

前記キャッシュメモリ上に格納された命令を実行する命令実行手段と、

命令の実行状態を記憶する複数のレジスタと、

前記第２の復号手段によって読み込まれた命令の実行中に割り込みが発生した場合に、割り込み発生時点のレジスタ値を、ランダムな値を持つ暗号化鍵で暗号化するレジスタ暗号化処理部と、

暗号化した前記レジスタ値をコンテキスト情報として前記プロセッサの前記外部メモリに保存する保存手段

とを具備することを特徴とするマイクロプロセッサ。

【請求項２】

前記レジスタ値の暗号化に用いた前記暗号化鍵を、前記公開鍵でさらに暗号化し、生成された暗号化鍵情報を前記外部メモリに出力する出力手段を更に備えることを特徴とする請

求項１に記載のマイクロプロセッサ。
【請求項３】
所定のレジスタによって指定されたアドレスの前記外部メモリから前記コンテキスト情報を読み込む読み込み手段と、
前記コンテキスト情報を、プログラム単位にプロセッサが保持する復号鍵で復号して前記レジスタに書き込む復号手段と、
前記復号されたコンテキスト情報に保持された再開アドレスに基づいて、所定の鍵テーブルから復号鍵情報を読み出し、プロセッサの秘密鍵により復号してプログラムの復号鍵を取得する復号鍵取得手段と、
前記再開アドレスから読み込まれた暗号化されたプログラムを、前記プログラムの復号鍵で復号して実行する実行手段
とを更に備えることを特徴とする請求項１に記載のマイクロプロセッサ。
【請求項４】
前記第２の復号手段によって復号化された前記暗号化プログラムの実行中に発生した割り込みが、前記暗号化プログラムのソフトウェア割り込み命令の実行によるものであることを検出する検出手段を更に備え、前記レジスタ暗号化処理部は、前記割り込みが前記暗号化プログラムのソフトウェア割り込み命令の実行によるものである場合には、前記複数のレジスタ中の所定のレジスタを暗号化対象から除外してレジスタ暗号化処理を行うことを特徴とする請求項１に記載のマイクロプロセッサ。
【請求項５】
所定の外部メモリアドレスから暗号化鍵情報を読み込み、マイクロプロセッサに埋め込まれた秘密鍵により復号してレジスタ値の暗号化鍵を取得する復号手段と、
所定の外部メモリアドレスから前記コンテキスト情報を読み込み、前記暗号化鍵に基づいて復号して前記レジスタに書き込む復号手段と、
前記復号されたコンテキスト情報に保持された再開アドレスに基づいて、所定の鍵テーブルから復号鍵情報を読み出し、プロセッサの秘密鍵により復号してプログラムの復号鍵を取得する復号鍵取得手段と、
前記再開アドレスから読み込まれた暗号化されたプログラムを、前記プログラムの復号鍵で復号して実行する実行手段
とを更に備えることを特徴とする請求項２に記載のマイクロプロセッサ。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、パーソナルコンピュータなどのハードウエアに組み込まれて暗号化されたプログラムを実行するマイクロプロセッサ、このマイクロプロセッサを組み込んだハードウエア装置にインターネット等の公衆ネットワークを通じて暗号化された実行プログラムを配布するプログラム配布システムと配布方法、および配布された実行プログラムをハードウエア内の既存のオペレーティングシステムと整合してロードすることのできる暗号化ファイルロード方法に関する。
【０００２】
【従来の技術】
近年、マイクロプロセッサの性能向上は著しく、従来の機能である計算やグラフィックだけではなく、動画像や音声の再生、編集加工が可能になっている。このようなマイクロプロセッサをエンドユーザ向けシステム（以下、「ＰＣ」と呼ぶ。）に組み込むことによって、ユーザはモニタ上でさまざまな動画像や音声を楽しめるようになった。また、動画像や音声の再生機能を、ＰＣ本来の計算能力と組み合わせることによって、ゲームなどへの応用を高めることができる。このようなマイクロプロセッサは固有のハードウエアを必要とするわけではなく、さまざまなハードウエアに組み込まれ得るので、すでにＰＣを持っているユーザは、プログラムを実行するマイクロプロセッサを取り代えるだけで、安価に動画像や音声の再生、編集が楽しめるという利点もある。

【０００３】
ＰＣで画像や音声を扱う場合に問題となるのが、オリジナルの画像や音楽の著作権の保護である。ＭＤやディジタルビデオデッキでは、これらの装置に不正なコピーを防ぐための機構をあらかじめ組み込むことによって、無制限なコピーを防止することができる。これらの装置を分解、改造してまでも不正コピーを行うことは極めて稀であり、仮にそのような装置があったとしても、世界的にみて、不正コピーを目的として改造された装置の製造、販売を法律によって禁じられる方向にある。したがって、ハードウエアの面での不正コピーによる被害は、それほどの問題とはなっていない。
【０００４】
しかし、ＰＣ上で画像データや音楽データを扱うのは、ハードウエアではなく、ソフトウエアそのものである。そして、エンドユーザはＰＣ上でソフトウエアの改変を自由に行うことができる。すなわち、ユーザにある程度の知識があれば、プログラムを解析することにより、実行ソフトウエアを書き換えて、不正コピーを行い得る可能性は充分にある。さらに、このようにして作られた不正コピー用のソフトウエアは、ハードウエアと異なり、ネットワークなどの媒体を通じて、またたく間に広がってしまうという問題がある。
【０００５】
このような問題を解決するために、著作権が問題となる商業映画や音楽の再生に使われるＰＣソフトウエアには、ソフトウエア自体を暗号化するなどの手法により、解読、改竄されることを防止する技術が用いられている。この技術は、耐タンパソフトウエア技術と呼ばれている（David Aucsimith, et al.;"Tamper Resistant Software: An Implementation", Proceeding of the 1996 Intel Software Developer's Conference）。耐タンパソフトウエア技術は、動画像や音声を始めとしてＰＣを通じてユーザに提供される著作物や、ノウハウとして価値のある情報が不正にコピーされるのを防止するだけではなく、ＰＣソフトウエア自体に含まれるノウハウなどを解析から守るのにも有効である。
【０００６】
【発明が解決しようとする課題】
しかしながら、耐タンパソフトウエア技術の根本原理は、プログラムの中で保護を要する部分を実行前は暗号化しておき、実行する直前に復号し、実行終了時に再び暗号化することにより、逆アセンブラ、デバッガなどの解析ツールによる解析を困難にするものである。したがって、プログラムがプロセッサによって実行可能である以上、プログラム開始時から順を追って解析していけば必ず解析することが可能である。換言すれば、耐タンパソフトウエア技術を用いても、プログラムコードを逐次的に解析することによって、プログラムの動作が解読可能となる。
【０００７】
この事実は、ＰＣを利用して動画像や音声を再生するシステムに、著作権者が著作物を供給する際の妨げとなる。また、ＰＣを通じて高度な情報サービスや企業、個人のノウハウを含んだプログラムをＰＣに適用しようにも、不正コピーの可能性が依然存在する以上、適用が見合わせられ、ＰＣソフトウエアの応用範囲が著しく狭められるという問題もある。
【０００８】
そこで本発明の第１の目的は、実行プログラムの書き換えを確実に防止することができ、ＰＣソフトウエアの広範囲な適用を可能にするマイクロプロセッサの提供にある。
【０００９】
本発明の第２の目的は、必要な暗号化回路、復号化回路、メモリのコストを削減し、安価かつ高性能なマイクロプロセッサの提供にある。
【００１０】
本発明の第３の目的は、ネットワークを介して、クライアント側装置に安全にプログラムを配布することのできるプログラム配布装置の提供にある。
【００１１】
本発明の第４の目的は、ネットワークを介して配布されたプログラムを安全に受信するこ

とのできるクライアント側装置の提供にある。
【００１２】
本発明の第５の目的は、ネットワークを介して安全にプログラムを配布することのできるプログラム配布システムの提供にある。
【００１３】
本発明の第６の目的は、公衆ネットワークを介した安全なプログラムの配布方法の提供にある。
【００１４】
本発明の第７の目的は、暗号化されたプログラムを、オペレーティングシステムとの不整合なくロードすることのできるプログラムロード方法の提供にある。
【００１５】
本発明の第８の目的は、第三者による不正な解析から保護することを可能にした、実行ファイルを記録したコンピュータ読み取り可能な記録媒体の提供にある。
【００１６】
【課題を解決するための手段】
上記の第１の目的を達成するために、本発明のマイクロプロセッサは、外部へ読み出すことのできない固有の秘密鍵を内部に保持し、この秘密鍵に対応する公開鍵であらかじめ暗号化された内容を復号化する１チップまたは１パッケージのマイクロプロセッサを提供する。このマイクロプロセッサは、仮想アドレスを物理アドレスに変換するアドレス変換手段と、このアドレス変換手段の変換規則と各々が前記仮想アドレスで指定された範囲の暗号化属性情報を有する１以上のエントリとを含むテーブルを格納する第１の記憶手段（たとえば変換索引バッファ）と、マイクロプロセッサ外部のメモリから指定された物理アドレスに対応する内容を読み出すメモリ読み出し手段と、前記テーブルのエントリに含まれる暗号化属性情報が、仮想アドレスで指定された範囲が暗号化されていることを示す場合に、外部メモリに記録された内容をメモリ読み出し手段を介して読み出し、前記固有の秘密鍵によって復号化する復号化手段と、復号化手段で復号された内容を一時的に記憶する第２の記憶手段と（たとえば一次キャッシュ）、第２の記憶手段に記憶された内容を逐次解釈する命令デコード手段とを備える。アドレス変換手段と、メモリ読み出し手段は、たとえばバスインターフェイスとして実現され得る。
【００１７】
復号化され、第２の記憶手段に格納された内容は、命令デコード手段以外からは読み出し禁止とされる。また、仮想アドレスで使用している外部メモリの対応する物理アドレスに対して読み出しが行われた場合には、この読み出しに対して例外処理を発生させるか、または物理アドレスに記録されている値とは異なる値を返す。このように、暗号化された内容の復号化処理は、マイクロプロセッサに固有の秘密鍵によって、完全にマイクロプロセッサ内部で行われ、かつ復号化されたデータの外部からの読み出しが防止される。すなわち、ＰＣの所有者によるプログラムの解析、改変が効果的に防止される。また、暗号化属性を示す情報、たとえば暗号化フラグをエントリに設定することにより、既存のプログラムにソース変更や再コンパイルなどの変更を加えることなく、暗号化フラグの状態を見ることによって、暗号化されたプログラムを滞りなく実行することが可能になる。
【００１８】
また、暗号化プログラムの実行中の割り込みの発生を考慮して、マイクロプロセッサは、復号化された内容の実行状態を表わす値を保持するレジスタと、暗号化された内容の実行中に割り込みが発生した場合に、レジスタに記憶されている値を、ランダムな値を持つ任意の暗号化鍵で暗号化するレジスタ暗号化／復号化処理部とをさらに有する。暗号化に用いた任意の暗号化鍵自体を、マイクロプロセッサに固有の秘密鍵でさらに暗号化して、レジスタに保存する。割り込みが終了して、暗号化プログラムを続行する場合は、レジスタ値暗号化／復号化処理部は、レジスタの内容を復号化する。この構成により、不正な解析をより困難にし、安全性を高めている。
【００１９】

本発明の第２の目的を達成するために、１チップまたは１パッケージのマイクロプロセッサは固有の秘密鍵とこれに対応する公開鍵を有し、公開鍵に代えて、任意の共通鍵を用いて暗号化された暗号化プログラムを実行する。暗号化に用いられた任意の共通鍵は、マイクロプロセッサに固有の公開鍵によってあらかじめ外部で暗号化され、マイクロプロセッサに保持される。このマイクロプロセッサは、仮想アドレスを物理アドレスに変換するアドレス変換手段と、アドレス変換手段の変換規則と公開鍵によって暗号化された共通鍵についてのキー情報とを含むキーテーブルを格納する第１の記憶手段（たとえば命令ＴＬＢ）と、マイクロプロセッサ外部のメモリに対して、指定した物理アドレスに対応する内容を読み出すメモリ読み出し手段と、暗号化された共通鍵を秘密鍵で復号する第１の復号化処理手段と、暗号化されたプログラム内容を、復号化された共通鍵で復号する第２の復号化処理手段と、復号化されたプログラム内容を一時的に記憶する第２の記憶手段（たとえば一次命令キャッシュ）と、第２の記憶手段に記憶された内容を逐次解釈する命令デコード手段とを備える。

【００２０】

このマイクロプロセッサでは、任意の共通鍵（対象鍵）によってあらかじめ暗号化されたプログラムを復号化するには、まず、マイクロプロセッサに固有の公開鍵で暗号化された共通鍵を、対応する秘密鍵で復号化しなければならない。したがって、固有の秘密鍵を有するユーザでないと、暗号化プログラムを復号化することができない。一方、プログラム自体は、非対称鍵アルゴリズムよりもずっと簡単な対象鍵（共通鍵）アルゴリズムで暗号化されているので、共通鍵さえ復号できれば、プログラムの復号化は簡単にできる。これによりハードウエアのコストが低減され、復号化の速度が向上する。

【００２１】

キー情報は、キーテーブルを参照して、複数のエントリで共有される。これにより、鍵の保持に必要な高速メモリの量を低減することができる。

【００２２】

第３の目的を達成するために、上記の第１および第２の目的と関連して説明したマイクロプロセッサが内蔵されたクライアント装置に、ネットワークを介して実行プログラムを配布するプログラム配布装置を提供する。上述したように、マイクロプロセッサはあらかじめ固有の公開鍵と秘密鍵とを有する。このプログラム配置装置は、クライアント装置との間に第１の通信路を設定する第１通信路設定部と、前記第１の通信路を介して前記クライアント装置を使用するユーザの認証を行うユーザ認証部と、前記第１の通信路上に、クライアント装置が内蔵するマイクロプロセッサに直接連絡する第２の通信路をさらに設定する第２通信路設定部と、クライアント装置に配布すべき実行プログラムを暗号化して暗号化プログラムを作成する暗号化処理部と、暗号化したプログラムを前記第２の通信路を介してクライアント装置のマイクロプロセッサに送信する配布送信部とを備える。

【００２３】

暗号化処理部は、第２の通信路を介してクライアント装置のマイクロプロセッサから送られてくる固有の公開鍵を用いてプログラムを暗号化する。あるいは、任意の暗号化鍵でプログラムを暗号化し、この暗号化鍵を、クライアント装置のマイクロプロセッサから送られてくる公開鍵で暗号化する。暗号化された暗号化鍵は、暗号化されたプログラムと共にクライアント装置に配布される。

【００２４】

第４の目的を達成するために、ネットワークを介してプログラム配布装置からプログラムの配布を受けるクライアント装置は、上述のあらかじめ固有の秘密鍵と公開鍵とを有するマイクロプロセッサと、前記プログラム配布装置との間に第１の通信路を設定する第１クライアント側通信路設定部と、第１の通信路を介して、前記クライアント装置を使用するユーザのユーザＩＤを前記プログラム配布装置に送信するユーザ認証部と、第１の通信路上に、前記マイクロプロセッサから直接プログラム配布装置に連絡する第２の通信路をさらに設定する第２クライアント側通信路設定部と、前記第２の通信路を介してプログラム配布装置に、マイクロプロセッサが固有の秘密鍵と公開鍵とを確かに保持することを証明

する証明を送信する証明部と、第２の通信路を介して前記プログラム配布装置から、暗号化された実行プログラムを受信する受信部とを有する。

【００２５】

第５の目的を達成するために、本発明のプログラム配布システムは、ネットワークと、このネットワークに接続された上述したプログラム配布装置と、ネットワークに接続された上述したクライアント装置とを含む。

【００２６】

第６の目的を達成するために、本発明のプログラム配布方法は、固有の秘密鍵と、この秘密鍵に対応する固有の公開鍵とをあらかじめ有するマイクロプロセッサを内蔵するコンピュータから、プログラム配布装置に、ネットワークを介して第１の通信路を設定する。この第１の通信路上に、前記マイクロプロセッサからプログラム配布装置に直接連絡する第２の通信路をさらに設定する。第２の通信路を介して、前記マイクロプロセッサから前記プログラム配布装置に、前記固有の公開鍵を送信する。そして、ネットワークを介して送信された公開鍵を、プログラム配布装置で受信し、プログラム配布装置でプログラムを暗号化する。暗号化したプログラムを、第２の通信路を介して、コンピュータのマイクロプロセッサに直接送信する。

【００２７】

このようにして配布されるプログラムは、コンピュータが内蔵するマイクロプロセッサに固有の公開鍵で暗号化されているか、あるいは共通鍵で暗号化され、この共通鍵をマイクロプロセッサに固有の公開鍵で暗号化してある。プログラムは、コンピュータを介さずに、直接マイクロプロセッサに送信され、この公開鍵と対応する固有の秘密鍵を有するマイクロプロセッサでなければ復号することができない。このような配布方法により、第三者による不正な復号を効果的に防止することができる。

【００２８】

第７の目的を達成するために、本発明のプログラムロード方法は、暗号化されたプログラムに対して１以上のページからなる仮想記憶領域を割り当て、前記暗号化されたプログラムを前記１以上のページに書き込んでから、前記１以上のページのそれぞれに対応するページテーブルエントリの暗号化フラグをセットする。これにより、既存のオペレーティングシステムとの不整合を生じさせることなく、暗号化されたプログラムを実装することができる。また、このようなロード方法を実行させるプログラムをあらかじめコンピュータ読み取り可能な記録媒体に記録してもよい。ここでいう記録媒体とは、たとえばフロッピーディスク、ＣＤ－ＲＯＭ、ＭＯディスクなどのコンピュータ外部のメモリ装置、半導体メモリ、磁気ディスク、光ディスク、磁気テープなどを含む。

【００２９】

さらに、第８の目的を達成するために、暗号化されたプログラムファイルを記録するコンピュータ読み取り可能な記録媒体を提供する。このプログラムファイルは、外部のプログラムへの直接の参照を含まない１以上の第１のプログラム領域と、外部のプログラムへの直接の参照を含む１以上の第２のプログラム領域を有する。第１プログラム領域は、あらかじめ定められた暗号化鍵で暗号化された実行ファイルを記録し、第２プログラム領域は、外部のプログラムモジュールへのジャンプ命令を並べたアドレス変換用のテーブルを平文の状態で記録する。このプログラムファイルがプログラムローダによりコンピュータのメモリ上に読み込まれるとき、外部への参照を含む第２領域は、そのコンピュータの外部プログラムの配置に基づいて適切なアドレスを参照するようにプログラムローダによって書き換えられる。このとき、第２領域が平文の状態でアドレス変換テーブルを格納しているので、暗号化された実行プログラムをロードするときにリロケーションで不整合が生じない。また、ロード時の不整合を防止すると同時に、暗号化されてデータを格納する第１領域により、プログラムの大部分を不正な解析から保護することが可能になる。

本発明のその他の特徴、効果は、以下に述べる実施の形態によって、より明確になるものである。

【００３０】

【発明の実施の形態】
以下、図面を参照して本発明を詳細に説明する。
【００３１】
＜第１実施形態＞
図１～図４は、本発明の第１実施形態に係るマイクロプロセッサ１０を説明するための図である。図１の概略ブロック図に示すように、マイクロプロセッサ１０は、このマイクロプロセッサに固有の秘密鍵１１２と、この秘密鍵とペアをなす公開鍵１１４と、この公開鍵によりあらかじめ暗号化された実行プログラムを実行する際に秘密鍵１１２で復号化する復号化処理部１１６と、復号化されたプログラムを一時的に格納する一次命令キャッシュ１１８と、一次命令キャッシュ１１８に格納された内容を逐次解釈する命令フェッチ／デコード手段２３０と、バスインターフェイスユニット１２２と、命令ＴＬＢ１２０とを、１チップ上または１パッケージ１３５内に備える。
【００３２】
バスインターフェイスユニット１２２は、仮想アドレスを物理アドレスに変換するアドレス変換手段（不図示）と、マイクロプロセッサ外部のメインメモリ２０３から、指定された物理アドレスの内容を読み出すメモリ読み出し手段（不図示）とを有する。また、命令ＴＬＢ１２０は、アドレス変換手段の変換規則と、各々が仮想アドレスで指定された範囲が暗号化されているかどうかを示す暗号化フラグを有する１以上のエントリとを含むページテーブルを読み込む。
【００３３】
マイクロプロセッサ１０は、たとえばＰＣに組み込まれると、システムバス１０７を介して外部記憶装置１０１、ＰＣのメインメモリ１０３、２次キャッシュ１０５に接続される。ユーザは、所望のプログラムを暗号化された状態で購入したい場合に、このマイクロプロセッサ１０に固有の公開鍵を、ＰＣを介してプログラムベンダに送信する。プログラムベンダは、送られてきた公開鍵でプログラムを暗号化して、暗号化された実行プログラムをユーザのＰＣに送り返す。暗号化された実行プログラムは、たとえばＰＣのメインメモリ１０３に図２に示すファイル形式でロードされ、実行する場合にだけマイクロプロセッサ１０の復号化処理部１１６で復号化され、実行される。このとき、マイクロプロセッサ１０に固有の秘密鍵１１２を用いなければプログラムの復号化はできない。
【００３４】
このマイクロプロセッサ１０は仮想記憶性能を有する。したがって、図３に示すように、メインメモリ１０３の所定の範囲を指定する物理アドレスによって区切られたページ３８－１～３８－ｋと、これらに対応するエントリを有するページテーブル３７が設定される。命令ＴＬＢ１２０は、マイクロプロセッサが実行しようとする範囲のプログラムが書き込まれた物理ページに対応するページテーブルエントリを取り込む。各エントリには、上述したように暗号化フラグが設けられており、対応するページの内容が暗号化されている場合に暗号化フラグがセットされる。復号化処理部１１６は、エントリに暗号化フラグがセットされている場合にのみ、このエントリに対応するページの内容を復号化する。ページテーブルと暗号化フラグの詳細については、後述する。
【００３５】
暗号化プログラムの実行中は、復号化されて一次命令キャッシュ１１８に格納されていたプログラムの命令は、順次、命令フェッチ／デコード部１３０によってデコードされ、命令プール１３１に供給される。命令実行切り替え部１３２は、一次データキャッシュ１３４から順次データをロードする。命令の実行が終了すると、命令実行完了部１３３を介して、メモリに書き込まれるべきデータは一次データキャッシュ１３４に書き戻される。
【００３６】
このような暗号化プログラムの実行中に、割り込みによって復号化が中断されることがあり得る。中断が起きると、従来のマイクロプロセッサでは、別のタスク（コンテキスト）への切り替えが発生するときに、プロセッサの内部状態をメインメモリ１０３のスタック上に書き出す。割り込み中のタスクの処理が終了したら、メインメモリ１０３に書かれた

内容が復元され、実行が再開される。しかし、プロセッサの実行コンテキスト情報を調べれば、そこにはマイクロプロセッサの実行アドレスと内部状態に関する情報が含まれるため、プログラムがどのような動作をしていたかがわかってしまう。

【００３７】

これを防止するために、本発明のマイクロプロセッサ１０は、レジスタ値暗号化／福号化処理部１２６を有する。レジスタ値暗号化／福号化処理部１２６は、暗号化プログラムの実行中に割り込みや例外が発生した場合に、実行済みの命令の状態を保持しているレジスタ１２４の内容を、一旦暗号化する。このときの暗号化の鍵は、マイクロプロセッサ１０がランダムな値を割り当てたものである。

この暗号化鍵をさらにマイクロプロセッサ１０の固有の公開鍵２１４で暗号化して、暗号化されたコンテキスト情報と共にスタック上に保存する。ただし、コンテキスト情報のうち、オペレーティングシステムがそのプロセスを再開するのに必要な次レジスタ情報は暗号化されない。また、プログラムの再開アドレス、スタックポインタなどのシステムレジスタと、例外発生ページアドレスは暗号化されない。これら以外の汎用レジスタが暗号化される。

【００３８】

割り込みが終了し、リターン命令によってもとの暗号化プログラムに戻る場合は、マイクロプロセッサ１０はスタック上の暗号化されたレジスタ値を一度レジスタ１２４に読み込む。レジスタ値暗号化／福号化処理部１２６は、レジスタ１２４上に読みこまれた暗号化値を復号化し、実行を再開する。より具体的には、マイクロプロセッサ１０は、保存されたコンテキストの再開アドレスを調べる。再開アドレスが暗号化されたページであれば、レジスタ値暗号化／復号化処理部１２６は、秘密鍵１１２を用いてコンテキスト情報を復号化する。このような割り込み時、および復帰時のレジスタの暗号化、復号化は、いかなるプログラムからも知ることはできない。

【００３９】

なお、システムコール呼出しに使われるソフトウエアインタラプトの場合は、パラメータの受け渡しに汎用レジスタが使われるため、この汎用レジスタについては暗号化を行わない。すなわち、実行中の暗号化プログラム自体が発生させる割り込み（ソフトウエアインタラプトと呼ばれ、ユーザモードからＯＳのカーネルモードに制御を移すときに使われる割り込み処理）では、レジスタ１２４の暗号化は行わない。ソフトウエアインタラプトはシステムコール発行に伴って実行されるので、暗号化プログラムの作成者はあらかじめこの点に留意したうえで、セキュリティが守られるようにプログラムを作成する。

【００４０】

暗号化プログラムの実行が終了すると、命令ＴＬＢ１２０に読み込まれたエントリにセットされていた暗号化フラグがクリアされ、一次命令キャッシュ１１８に格納された平文状態のプログラムデータは破棄される。このとき、復号化された内容がメインメモリ１０３に書き戻されることはない。これにより、完全にマイクロプロセッサ１０の内部だけで、暗号化された実行プログラムの復号が行われ、他のソフトウエアもユーザも、復号化を知ることができない。

【００４１】

図２は、第１実施形態における実行プログラムのファイル形式を示し、図３は、第１実施形態のマイクロプロセッサ１０における論理アドレス３１と、仮想メモリを管理するページテーブル３７との関係を示す。図４は、図３に示すページテーブルのうち、エントリ３７－ｊの構成を示す。

【００４２】

図２に示すように、第１実施形態のプログラムファイルは、このプログラムの外部のプログラムへの直接の参照を含まない第１のプログラム領域（すなわち、.textセクション、.bssセクション、.rdataセクションなど）と、このプログラムの外部のプログラムへの直接の参照を含む第２のプログラム領域（すなわちＩＡＴセクション）とを有する。第１のプログラム領域には、データはあらかじめ暗号化鍵で暗号化された状態で記録され、第２

のプログラム領域には、平文の状態でジャンプテーブルが記録される。

【００４３】

ファイルの先頭には、ＭＳ－ＤＯＳヘッダ１１およびＣＯＦＦファイルヘッダ１２があり、プログラム全体の属性などを定義する。プログラムは、.textセクション１３、.bssセクション１４など、いくつかのセクションに分かれており、各セクションごとに、属性を定義するセクションヘッダと、内容が含まれるセクション本体がある。通常、プログラムのコード領域は.textセクションに格納される。ただし、図２に示すように、ＩＡＴ（Import Address table）と呼ばれる外部プログラムへのジャンプ命令が格納される領域は、コード領域ではあっても、プログラム本体とは別の.IATセクションに格納される。セクションヘッダにはいくつかのフィールドがあり、その中の特性フィールド１３－１が、セクションのメモリ上の配置境界や読み書き属性などを指定している。

【００４４】

第１実施形態では、特性フィールド１３－１上の１ビット、たとえば０ｘ００００００４００でマスクされる１ビットが、暗号化属性を指定する暗号化フラグとして用いられる。このビットが１であれば、対応のセクションが暗号化されていることを示し、０であれば暗号化されていないことを示す。図２の例では、このビットが値１を有するので、対応のセクションは暗号化されていることになる。たとえば、.textセクション１７の内容は、本来平文であったプログラムをマイクロプロセッサ１０の固有の公開鍵１１４で暗号化されたものである。一方、ジャンプテーブルが格納される.IATセクションヘッダ１６の暗号化フラグは値０を持ち、このセクションにはジャンプテーブルが暗号化されず、平文のまま格納されることを示す。

【００４５】

一般に、プログラムがロードされるアドレスは、システム構成や、そのメモリの利用状況によって異なる。そのようなとき、プログラムに含まれる参照アドレスをプログラムローダが状況に合わせて変更する必要が生じる。これをリロケーションと呼ぶ。あるプログラム内部に閉じた関数呼出しは、プログラム内部の相対的なアドレスに基づいてアクセスを行うリロケータブルなコードを生成することによって、リロケーション処理を不要にすることができる。しかし、プログラム外部のモジュールの呼出し、たとえばシステムコール呼出しや、その逆に外部のモジュールからプログラム内部の関数が呼び出される場合には、モジュールの絶対アドレスも相対的な位置関係も、プログラムがロードされるまでわからないことが多い。システムのハードウエア構成やソフトウエアのバージョンによってプログラムや作業領域のサイズが変わることがその理由である。

【００４６】

この問題を解決するため、次のような手法を用いる。外部関数への呼出しを含むプログラムファイルに、ＩＡＴ（Import Address Table）と呼ばれる、読み出される外部関数へのジャンプ命令を並べたテーブルをあらかじめ作成しておき、暗号化されたプログラム本体（ここでは.textセクション１７）から外部モジュールへの呼出しは、このテーブルへのコール命令として間接的に行う。ジャンプテーブルには、呼出し先の関数を識別する名前が付加されており、プログラムローダはプログラムのロード時に名前に基づいて外部関数のアドレスを検索し、上記テーブルの対応する命令のジャンプ先をその関数のアドレスに書き換える。この状態でプログラムから外部関数への呼出しが行われると、制御は一度ＩＡＴのジャンプ命令に移り、次にジャンプ命令の飛び先の目的とする関数に制御が移る。第１実施形態では、.IATセクション２０にこのジャンプテーブルが格納されている。

【００４７】

ＩＡＴ領域はプログラムローダによる書き換えが行われるので、この領域が暗号化されていると、書き換えが正常に行われなくなり、正しい外部関数が呼び出されないことになる。そこで本発明では、図２に示すように、ＩＡＴをプログラム本体（.textセクション１７）とは別のセクションに設け、.IATセクション２０を平文のまま維持することにより、リロケーション処理が正しく行われる。ＩＡＴに含まれる情報は、外部関数への呼出しだけなので、プログラム自体の秘密が損なわれることはない。

【００４８】
第１実施形態では、ＩＡＴを外部関数へのジャンプ命令が格納されていることとしたが、プログラム本体からの呼出し命令が、ＩＡＴの内容で指定されるような間接呼出し命令を使っている場合には、ＩＡＴの内容はジャンプ命令ではなく、単なるとび先のアドレスが格納されたデータ列であってもよい。もちろんこの場合も、ＩＡＴは暗号化されない。
【００４９】
次に、ページテーブルエントリに設けられた暗号化フラグについて説明する。暗号化された実行プログラムは、ローダプログラムによってメインメモリ１０３上に読み込まれる。このとき、ローダは、ロードするプログラムのために１以上のページから成る仮想メモリ空間を割り当て、これらのページにプログラムのすべてのセクションを書き込んで配置する。配置が終わると、ローダは.textセクション１７が配置されたページテーブルエントリ37-jの暗号化フラグ37-j-E（図４）を１にセットする。このページテーブルエントリ37-jが、マイクロプロセッサ１０の命令ＴＬＢ１２０にキャッシュされているときに暗号化フラグがセットされていると、バスインターフェイスユニット１２１がそれを監視していて、一次命令キャッシュ１１８と二次命令キャッシュ１０５の、このページエントリに対応するキャッシュエントリを無効化して、復号化された内容を記憶するためのスペースをあける。このとき、平文状態の.IATセクションが配置されたページテーブルエントリの暗号化フラグは０のままである。
【００５０】
いったん、ページテーブル307-j-Eの暗号化フラグがセットされると、以後このページテーブルに対応する物理メモリのページ38-kへの他プログラムからのアクセスは禁止される。すなわち、このページへの読み出し、書き込みは一切できなくなる。この保護は、実行がユーザモード、カーネルモードのどちらの場合にも適用される。
【００５１】
暗号化フラグがセットされている間に、外部から読み書きの要求があった場合は、保護例外が発生する。あるいは、読み出しに対しては、常に所定の値（たとえば０）もしくは乱数が読み出され、書き込みに対しては無効とするようにしてもよい。例外の扱いについては後述する。暗号化フラグがセットされた物理ページ310-k が平文状態で一次データキャッシュ１３４にキャッシュされることはない。以下、ページテーブルの暗号化フラグがセットされたページに対応するメモリ領域を暗号化ページまたは暗号化領域と呼ぶ。暗号化領域の実行では、プロセッサのデバッグ機能、たとえばステップ実行機能などは無効化される。
【００５２】
ローダは、メインメモリ１０３上にページテーブルを設定すると、プログラムのエントリポイントに制御を移し、プログラムの実行が開始される。マイクロプロセッサ１０が命令として暗号化領域にアクセスすると、キャッシュのミスヒットが生じ、メインメモリ１０３から一次命令キャッシュ１１８への読み込みが始まる。このとき、暗号化領域については、復号化処理部１１６が秘密鍵１１２を使用して内容を復号化し、一次命令キャッシュ１１８上には、実行可能な命令列が読み込まれ、マイクロプロセッサ１０はそれを実行する。暗号化領域以外をキャッシュする場合には、復号化処理部１１６はなにもせずに、メインメモリ１０３の内容はそのまま一次命令キャッシュ１１８にコピーされる。
【００５３】
上述したように、この復号化はすべてマイクロプロセッサ１０内部のハードウエアである一次命令キャッシュ１１８の動作の一部として行われる。したがって、前述したページテーブルの扱いを除けば、ソフトウエアは復号化操作を意識することはまったくない。ページテーブルの設定についても、通常はオペレーティングシステムによって設定が行われるので、アプリケーションからはまったく見ることができない。さらに、これも上述したように、マイクロプロセッサ１０に固有の秘密鍵１１２は、マイクロプロセッサ１０の内部に隠され、プログラムからはその値を読み出すことはできない。したがって、暗号化された実行プログラムの内容は、マイクロプロセッサ１０が組み込まれたＰＣのアプリケーシ

ョンプログラムはもちろん、オペレーティングシステムからも復号化された形で読み出すことは不可能である。

【００５４】

（ページの無効化）

プログラムの実行が終了すると、オペテーティングシステムは他のプログラムを動作させるために、そのプログラムの.textセクションが読みこまれていた暗号化領域のメモリページを開放して、別の内容の書き込みを認める必要がある。ページテーブルエントリに暗号化フラグがセットされたままでは、対応するページへの読み書きはできないので、ページテーブル上の暗号化フラグをリセットして、読み書きを許可する。仮想記憶を採用している場合は、プログラムの実行中にも、ページの一部がページアウトされて、ページの内容を二次メモリに書き出して、物理メモリを他の目的に使う必要が生じることがある。この場合も、ページの内容を読み出す前に、暗号化フラグはリセットされる。暗号化フラグがリセットされると、それを監視するバスインターフェイスユニット１２２は一次命令キャッシュ１１８の内容を無効化する。一次データキャッシュ１３４については、暗号化フラグがセットされた状態で一次データキャッシュ１３４への暗号化領域の読み込みが禁止されていれば、無効化の必要は特にない。

【００５５】

（メモリマップトファイル）

オペレーティングシステムの仮想記憶の実装によっては、プログラムの開始時にすべてのイメージを実際の物理メモリが読み込まれない場合もある。すなわち、プログラムの実行開始時にプログラムのイメージが仮想アドレスに割り付けられるだけで、物理メモリにロードされない場合もある。この仮想アドレスへの割付は、メモリマップトファイルと呼ばれる。通常、ページテーブルには、そのページが物理メモリに存在するかどうかを示すフラグがある。図４では、右下の37-j-Pがこのフラグに相当する。

【００５６】

通常、仮想メモリのあるページがアクセスされたとき、その仮想ページに物理メモリが割り当てられていない場合、オペレーティングシステムは例外原因を調べて、仮想ページに対応する物理ページを確保してからページテーブルを更新する。そして、ページ内容を二次メモリから読み出して、確保した物理メモリに書き込んでから、実行を再開する。対応のページが物理メモリに存在せず、かつ暗号化フラグがセットされている場合には、オペレーティングシステムは、まず物理メモリのスペースを確保してページテーブルを更新した後、ページ内容を書き戻すために、一度暗号化フラグをクリアしてから、通常の場合と同様に、二次メモリの内容を物理メモリに書き込み、その後再度暗号化フラグをセットして暗号化プログラムの実行を再開する。

【００５７】

（デバッグフラグ）

デバッグのためにプログラムの動作を検証するときには、そのプログラムが実行される実際の環境にできる限り近いことが望ましい。その意味では、暗号化プログラムのデバッグは、暗号化状態で行うべきである。しかし、プログラムが暗号化された保護状態では、ステップ実行などのデバッグ機能が使えず、コードを逆アセンブリすることもできないため、デバッグは事実上不可能に近い。このため、暗号化プログラムのデバッグは、どうしても平文状態で行う必要があると考えられる。

【００５８】

暗号化状態と平文状態での実行の差について考えると、マイクロプロセッサ１０でのコードの実行については、マイクロプロセッサ１０の内部では、プログラムが復号化された状態で実行されるため、暗号化された状態と平文の状態とで、動作に本質的な違いはない。しかし、メモリ保護については、暗号化されたプログラムを実行する場合、復号化されたプログラムを保護するため暗号化されたページへの読み出しを禁止している点で動作が異なっている。

【００５９】

そこで、暗号化プログラムのデバッグを、デバッグ作業の容易な平文状態で行い、平文状態で行い、かつデバッグ動作を暗号化状態と一致させるため、ページテーブルにデバッグフラグを設けている。図４に示すページエントリでは、37-j-Dがデバッグフラグに相当する。デバッグフラグがセットされているときに、そのページに対して暗号化プログラムの実行以外の読み書きが行われると、暗号化プログラムの保護例外が発生する。しかし、デバッグ作業は平文状態で行われているので、実行コンテキストは平文のままセーブされ、ステップ実行も可能となる。すなわち、プログラム実行においては、デバッグは平文のプログラムであっても、振る舞いは暗号化された状態と一致する。かつ、ステップ実行による実行トレースが可能になる。また、保護による例外が発生した場合も、セーブされた実行コンテキストを解析することにより、例外の発生原因を調べた上で、実行を再開することもできる。

【００６０】

ページテーブルについては、暗号化プログラムと平文のプログラムの間で暗号化フラグの値の相違は残るが、一般にアプリケーションプログラムはページテーブルの内容を意識することはないので、アプリケーションプログラムのデバッグの妨げにはならない。

【００６１】

暗号化プログラムのセキュリティを守るため、暗号化フラグがセットされた状態では、デバッグフラグをセットしても、ステップ実行などは禁止されたままにする。または、暗号化フラグがセットされた状態では、デバッグフラグをセットできないようにしてもよい。なお、暗号化フラグがセットされている限り、デバッグフラグの状態に関わらず、暗号化領域からのデータ読み出しは禁止される。

【００６２】

＜第２実施形態＞

図５～図８は、本発明の第２実施形態に係るマイクロプロセッサ２０を説明するための図である。第１実施形態では、マイクロプロセッサに固有の公開鍵を用いて暗号化されたプログラムを実行していた。しかし、公開鍵方式の暗号化／福号化アルゴリズムは、共通鍵方式に比較して一般に複雑で、回路規模が大きくなり、コストが高くなる上に、処理の高速化が困難である。そこで、第２実施形態のマイクロプロセッサは、共通鍵で暗号化されたプログラムを復号し実行するとともに、共通鍵を安全かつ効率的に管理する。

【００６３】

具体的には、マイクロプロセッサに供給される実行プログラムは、プログラムベンダが選択した任意の暗号化鍵（すなわち共通鍵）で暗号化される。暗号化に用いた共通鍵は、マイクロプロセッサに固有の公開鍵で暗号化される。共通鍵で暗号化された暗号化プログラムと、公開鍵によって暗号化された共通鍵とがマイクロプロセッサに送られてくるので、マイクロプロセッサは、固有の秘密鍵で共通鍵を復号化し、復号化された共通鍵で、暗号化プログラムの内容を復号化する。これにより、セキュリティを維持したまま、復号化のためのハードウエアコストを低減することができる。

【００６４】

図５に示すように、マイクロプロセッサ２０は、プログラムから読み出すことのできない秘密鍵と２１２、この秘密鍵に対応する公開鍵２１４と、この公開鍵であらかじめ暗号化された状態で外部のメモリに格納されている第３の暗号化鍵E_{Kp}［Ｋｘ］を前記秘密鍵２１２を使用して復号化する復号化鍵ＴＬＢ（第１の復号化処理手段）２３６と、外部のメモリに格納されている暗号化されたプログラムを、前記復号化された第３の暗号化鍵E_{Kp}［Ｋｘ］によって復号化する復号化処理部２１６（第２の復号化処理手段）と、復号化されたプログラムを格納する一次命令キャッシュ２１８と、バスインターフェイスユニット２２２と、命令ＴＬＢ２２０とを、１つのチップまたはパッケージ内に備える。

【００６５】

バスインターフェイスユニット２２２は、仮想アドレスを物理アドレスに変換するアドレス変換手段（不図示）と、マイクロプロセッサ外部のメインメモリ２０３から、指定された物理アドレスに対応する内容を読み出すメモリ読み出し手段（不図示）とを有する。命

令ＴＬＢ２２０は、アドレス変換手段による仮想アドレスから物理アドレスへの変換規則と、公開鍵によってあらかじめ外部で暗号化された第３の暗号化鍵Ｅ［Ｋｘ］（共通鍵）についてのキー情報とを含むテーブルを取り込む。より具体的には、命令ＴＬＢ２２０は、まずメインメモリ２０３のアドレスによって区切られたページ領域の各ページに対応するエントリを含むページテーブルを取り込む。各エントリは、対応するページの暗号化属性を示す暗号化フラグと、このページの内容を復号するための鍵情報が記憶されているキーテーブル上の位置を指定する識別子（キーエントリＩＤ）とを有する。この識別子を参照することによって、対応するキーテーブルのエントリを取り込む。

【００６６】

図６は、第２実施形態のマイクロプロセッサ２０で実行する暗号化プログラムを格納するファイル形式を示す。.textセクションヘッダ６３のフィールドには、暗号化アルゴリズム６３－２のフィールドと、暗号化鍵値６３－３のフィールドが追加されている。暗号アルゴリズム６３－２のフィールドには、コード領域を暗号化するための暗号化アルゴリズムが符号化されて格納される。暗号化鍵値６３－３のフィールドには、マイクロプロセッサ２０が暗号化プログラムを解読するために使う共通鍵Ｋｘが、マイクロプロセッサ２０に固有の公開鍵２１４で暗号化された値E_{Kp}［Ｋｘ］が格納されている。共通鍵Ｋｘの長さは、たとえば、暗号にトリプルＤＥＳを使った場合には１９２ビットが必要になる。そして、E_{Kp}［Ｋｘ］は、マイクロプロセッサ２０の公開鍵２１４によって暗号化される際のブロック長の整数倍のサイズとなる。

【００６７】

.textセクション６６は、共通鍵Ｋｘを用い、暗号化アルゴリズム６３－２のフィールドで指定されたアルゴリズムによって暗号化されている。ここでは、暗号化アルゴリズムには共通鍵方式のトリプルＤＥＳが指定されているものとする。特性フィールド６３－１には、プログラムが暗号化されていることを示すため、暗号化フラグが１にセットされていることは第１実施形態と同様である。

【００６８】

プログラムベンダが図２に示すような暗号化ファイルを作成するには、まず、コード領域の暗号化鍵を共通鍵Ｋｘとしてランダムに選び、.textセクション６６を、６４ビット（８バイト）ごとのブロック単位でトリプルＤＥＳアルゴリズムで暗号化する。暗号化鍵値６３－３には、解読用共通鍵Ｋｘをターゲットであるマイクロプロセッサ２０の公開鍵２１４で暗号化した値E_{Kp}［Ｋｘ］を格納し、暗号化アルゴリズムフィールド６３－２にはトリプルＤＥＳの符号表現を入れ、特性フィールドの暗号化フラグを１にセットする。

【００６９】

なお、コードを格納する領域を.textセクション６６に限らず複数のセクションとすることにより、同一プログラムのコードを複数の暗号化鍵もしくは暗号化アルゴリズムで暗号化してもよい。ただし、それぞれのセクションはプロセッサの仮想記憶（図７）のページ境界ごとに配置されなければならない。

【００７０】

また、あらかじめ、プログラムのコード領域の一部に、プログラムの動作と関係のない部分を設け、コード領域を暗号化する前に、その部分にプログラムを使用するターゲットのマイクロプロセッサ２０の公開鍵２１４やプログラムの配布先ユーザＩＤを直接、あるいは暗号化した形式で格納してもよい。こうすることによって,プログラムが不正解読され、不正コピーが配布された場合に、その経路を追跡することが可能になる。経路追跡のための情報は、プログラムの１サブルーチンとして実行可能な機械語命令列に符号化され、発見を困難にすることも可能である。

【００７１】

さて、暗号化されたプログラムをマイクロプロセッサ２０で実行する際には、実施形態１と同様に、まず、マイクロプロセッサ２０が組み込まれたＰＣのメインメモリ２０３に、暗号化プログラムをローダによって読み込むことから始まる。ローダは、暗号化プログラムのための仮想メモリ空間を含むプロセスコンテキストを設定し、すべてのセクションを

通常のプログラムを同様にメモリ上に配置し、必要に応じてリロケーションなどを行う。

【００７２】

図７は、第２実施形態に係るマイクロプロセッサ２０で暗号化プログラムを実行する際のページテーブルを示す。図７に示す第２実施形態のページテーブルはキーテーブル７９を有する拡張テーブルである。論理アドレス７１は、ディレクトリ７２、テーブル７３、オフセット７４の３つのフィールドに分割されている。論理アドレス７１を構成するこれら３つのフィールドは、次のようにして物理アドレスを計算する。

【００７３】

まず、最上位のディレクトリフィールドは、ディレクトリテーブル７６のエントリ76-iを指定する。ディレクトリテーブル７６の先頭は、マイクロプロセッサ２０のレジスタ７５によって指定される。ディレクトリエントリ76-iは、ページテーブル７７の先頭へのポインタを持つ。論理アドレス７１のテーブル７３によって、ページテーブル７７での位置が計算され、対応するページエントリ77-iにそのアドレスが示す実際の物理アドレス80-kが記述されている。ここまでは、第1実施形態と同様である。

【００７４】

図８は、図７に示したページテーブル７７のエントリと、対応するキーテーブル７９のエントリを示す。第２実施形態では、ページテーブル７７の各エントリに、対応するページを復号化するための鍵情報を指定する番号（識別子またはキーエントリＩＤ）を格納するフィールド77-j-Kが設けられる。この番号は、キーテーブル７９上の、復号化鍵が格納されているエントリの番号を指定する。キーテーブル７９の先頭はマイクロプロセッサ２０の制御レジスタのひとつであるキーテーブル制御レジスタ７８によって指定される。なお、各エントリは、第１実施例と同様に、対応するページの属性を示す暗号化フラグ77-j-Eと、デバッグフラグ77-j-Dも有する。

【００７５】

次に、ローダは暗号化されたプログラムを復号化するための共通鍵Ｋｘをマイクロプロセッサ２０に固有の公開鍵２１４で暗号化した値E_{Kp}［Ｋｘ］を格納するためのエントリ79-mを、キーテーブル７９の上に確保する。キーテーブル７９には、鍵情報と暗号化アルゴリズムのフィールドに加えて、参照カウンタのフィールド79-m-1があり、そのテーブルを参照するテーブルの個数が書き込まれる。すなわち、参照数が０であれば、そのテーブルは使用可能となる。ローダは使用するテーブルを決定して、キーテーブルの参照カウンタを０から１に書き換えてから、鍵情報と暗号化アルゴリズム情報をキーテーブルのフィールド79-m-3と79-m-4のそれぞれに書き込み、そのキーエントリの番号を対応するページテーブル７７のエントリ77-jに書き込む。同じキーエントリを参照するページが複数あれば、キーテーブル７９の参照カウンタを増やしてからページテーブル７７のエントリにキーエントリの番号を書き込む。

【００７６】

すでに同一のプログラムが実行されているなどの理由により、同一の鍵がキーテーブル７９に格納されている場合は、テーブルの確保は行わず、キーテーブルの参照カウンタを増やしてから、ページエントリにキーエントリの番号を書き込む。このようにして、暗号化鍵Ｅ［Ｋｘ］を複数ページで参照できるようにする。

【００７７】

上述のように、独立のキーテーブル７９を設けて複数のページで共通の復号鍵情報を共有できるようにした理由は、ページテーブルに必要なメモリの量を減らして命令ＴＬＢ２２０の効率を改善するためである。すなわち、第１実施形態では、暗号化プログラムの中のすべてのプログラムについて同一の公開鍵１１４が用いられていたため、各ページテーブルごとに鍵情報を格納する必要はなかったが、第２実施形態ではプログラムごとに必要な共通鍵の鍵情報が異なるため、ページごとに別々の鍵を格納する必要が生じる。しかし、比較的長いビット長を要する鍵情報を、直接ページテーブルのフィールドに割り付けると、ページテーブルの大きさが増大してメモリ２０３の利用効率が低下する。たとえば、ページが４Ｋバイトのときに、キャッシュラインサイズと同一の３２バイトの鍵をページテ

ーブルの各ページに割り当てると、ページサイズの１％弱のメモリがページテーブルのために必要となってしまう。そこで、ページテーブルの大きさを増大させたとしても、命令ＴＬＢ２２０のサイズが以前と同じであれば、命令参照に対する命令ＴＬＢ２２０のヒット数が低下し、プログラムの実行速度が低下する。命令ＴＬＢ２２０のサイズも同時に大きくすることも考えられるが、高速な論理－物理アドレス変換が要求される命令ＴＬＢ２２０の性質上、コスト高になってしまう。

【００７８】

また、実際には、暗号化鍵の種類は必ずしもページごとに変更する必要はなく、プログラムの秘密保護のためには、たかだか数種類の鍵が使い分けられれば十分であることが多い。そこで、第２実施形態では、複数のページが同一の復号化鍵情報を共有できるように独立のキーテーブル７９を設け、ページテーブル７７にはキーテーブル７９への番号を格納することにより、ページテーブルにル７７に必要なメモリ量を低減させたのである。

【００７９】

さて、プログラムの配置とページテーブルの鍵情報の設定が終わると、ローダは.textセクションヘッダ６３が配置されたページテーブルの暗号化フラグを１にセットする。そのページテーブルが命令ＴＬＢ２２０にキャッシュされている場合にページテーブルの暗号化フラグが１にセットされると、バスインターフェイスユニット２２２がそれを監視していて、一次命令キャッシュ２１８および二次命令キャッシュ２０５の対応するエントリを無効化する。

【００８０】

上述のキーテーブル７９の値は復号化鍵ＴＬＢ２６にも格納される。キーテーブル７９が書きかえられた場合は、対応する復号化鍵ＴＬＢ２３６のエントリも書き換えられる。ここで、復号化鍵ＴＬＢ２２０には、メモリ上の復号化鍵情報Ｅ［Ｋｘ］が、マイクロプロセッサ２０に固有の秘密鍵１１２で復号化されて格納される。このとき、メモリ２０３上の復号化鍵情報Ｅ［Ｋｘ］の長さを、復号化鍵ＴＬＢ２２０上の復号化された鍵情報Ｋｘより長く取ることにより、鍵の安全性とＴＬＢ２２０の利用効率を両立させることができる。マイクロプロセッサ２０内部の復号化鍵ＴＬＢ２２０に格納された鍵情報は、ユーザプログラムからは見ることができないからである。もちろん、ユーザプログラムによるキーテーブルの明示的なメモリ参照に対しては、復号化された値Ｋｘではなく、もとの暗号化されたままの値Ｅ［Ｋｘ］が返される。

【００８１】

キーテーブル７９の設定が完了すると、第１実施形態と同様に、ローダはプログラムのエントリポイントに制御を移し、プログラムの実行が開始される。プロセッサが命令として暗号化領域にアクセスすると、キャッシュのミスヒットが生じ、メインメモリ２０３から一次命令キャッシュ２１８への読み込みが始まる。このとき、ページテーブル７７からキーテーブル７９への参照で指定されたキーテーブル７９のエントリが、マイクロプロセッサ２０の秘密鍵２１２で復号化されて、復号化ＴＬＢ２３６に読み込まれる。復号化された鍵情報Ｋｘを使って、メインメモリ２０３に格納されていた暗号化データが復号化処理部２１６によって復号化され、一次命令キャッシュ２１８へと読み込まれ、実行される。

【００８２】

以後に行われる実行終了後のキャッシュの無効化、割り込み処理などは第１実施形態と同様である。

【００８３】

このように、第２実施形態ではプログラムの復号化に共通鍵方式を採用することができる。共通鍵のアルゴリズムは一般に同一鍵長さの公開鍵アルゴリズムと比較してハードウエア化が容易であり、復号化機能のコストを低減することができる。さらに、共通鍵をマイクロプロセッサに固有の公開鍵で暗号化し、その復号はマイクロプロセッサ２０に固有の秘密鍵でないとできないので、安全性も確保される。

【００８４】

＜第３実施形態＞

図９は、本発明の第３実施形態にかかるプログラム配布システム９０の図である。このプログラム配布システムは、基本的に、第１または第２の実施形態で説明したマイクロプロセッサを内蔵したコンピュータシステムにプログラムを配布するように設計されている。したがって、配布される実行プログラムは、マイクロプロセッサ固有の鍵で直接暗号化されるか、あるいは任意の共通鍵で暗号化され、共通鍵そのものををマイクロプロセッサ固有の鍵で暗号化して実行プログラムに添付する。このため、同一のプログラム媒体のコピー、たとえば大量にプレスされるＣＤ－ＲＯＭなどによってプログラムを配布することができない。そこで、ネットワークを介してプログラムを配布する。

【００８５】

図９に示すプログラム配布システム９０は、ネットワーク９５と、ネットワーク９５に接続され、実行プログラムをネットワーク９５を介して配布するプログラム配布装置９３と、同じくネットワーク９５に接続され、ネットワークを介してプログラム配布装置９３から実行プログラムの配布を受けるクライアント装置９１とを含む。

【００８６】

プログラム配置装置９３は、クライアント装置９１との間に第１の通信路を設定する第１通信路設定部９３１と、記第１の通信路を介してクライアント装置９１を使用するユーザのユーザ認証を行うユーザ認証部９３３と、第１の通信路上に、クライアント装置が内蔵するマイクロプロセッサに直接連絡する第２の通信路を設定する第２通信路設定部９３４と、第２の通信路を介して、クライアント装置９１のマイクロプロセッサ９０１が正当であることを認証するプロセッサ認証部９３５と、実行プログラムをプロセッサ９０１と対応付けられた暗号化鍵で暗号化して暗号化プログラムを作成するプログラム暗号化処理部９３６と、暗号化したプログラムを第２の通信路を介してクライアント装置に配布するネットワークインターフェイス９３１とを備える。

【００８７】

クライアント装置９１は、あらかじめ固有の秘密鍵と公開鍵とを有するマイクロプロセッサ９０１と、プログラム配布装置９３との間に第１の通信路を設定する第１クライアント側通信路設定部９０５と、第１の通信路を介してこのクライアント装置を使用するユーザのユーザＩＤを送信するユーザ認証部９１０と、第１の通信路上に、マイクロプロセッサから直接プログラム配布装置に連絡する第２の通信路を設定する第２クライアント側通信路設定部９０６と、第２の通信路を介して、このマイクロプロセッサが確かにこのマイクロプロセッサだけに固有の公開鍵と秘密鍵を有することを証明する証明をプログラム配布装置に送信する証明部９０７と、暗号化された実行プログラムを第２の通信路を介して受信するプログラム受信部９０８とを備える。

【００８８】

第２の通信路は、第１の通信路と同じ回線の異なるチャネル上に設定されてもよいし、第１の通信路と異なる回線上に設定されてもよい。暗号化されたプログラムは、マイクロプロセッサの認証が行われた上で、マイクロプロセッサに直接接続された第２の通信路を介してマイクロプロセッサに送信されるので、プログラム配布の安全性が確保されるとともに、ユーザによる不正を防止することができる。

【００８９】

図１０は、図９のプログラム配布システム９０におけるプログラム配布のシーケンス図である。

【００９０】

プログラム配布装置（サーバ）９３は、全世界一意となるような公開鍵と秘密鍵のペアＫ'p、Ｋ'sと、配布プログラムの原型である暗号化されていないプログラムファイルを有する。一方、クライアント装置９１はマイクロプロセッサ９０１を有し、マイクロプロセッサ９０１には、ネットワークを介してプログラムをダウンロードするための専用プログラムが添付されている。ダウンロードプログラムは、マイクロプロセッサ９０１の購入時、あるいはマイクロプロセッサ９０１を内蔵するシステムの購入時に添付されるか、あるいは郵送などのユーザ個別に配布が可能な任意の手段でよい。

【００９１】
ダウンロードプログラムには、マイクロプロセッサ９０１ごとに異なる固有の公開鍵Ｋｐ、証明書などの情報が埋め込まれている。具体的には、マイクロプロセッサ９０１の公開鍵Kpが実在するマイクロプロセッサに対応するものであることを示す証明書Certが、ダウンロードプログラムに格納されている。証明書を持つ目的は、ユーザが実際のマイクロプロセッサ９０１と対応しない既知の公開鍵と秘密鍵の組み合わせを使って暗号化されたプログラムを取得し、既知の秘密鍵で復号化することで平文プログラム内容を取得することを防ぐためである。証明書には、公開鍵Ｋｐと、Ｋｐを証明機関の鍵Ｋcertで署名した値$S_{Kcert}[Kp]$があらかじめ計算され、含まれている。Ｋcertの値は認証機関によって秘密に管理され、ダウンロードプログラムには含まれない。証明書に含まれる公開鍵Ｋｐと署名$S_{Kcert}[Kp]$を、認証用の公開鍵Ｋvalで検証することにより、Ｋｐが認証機関により正しく割り当てられたものであることが証明され、既知の鍵の組み合わせを使用できないようにしてある。

【００９２】
このような証明書は、マイクロプロセッサにあらかじめ組み込まれていてもよい。しかし、証明書に有効期限を設けた上でダウンロードプログラムに格納し、マイクロプロセッサに添付して販売するほうが、安全性がより高められる。この場合、定期的に更新された証明書を格納したダウンロードプログラムが、マイクロプロセッサの購買者に配布されることになる。

【００９３】
ダウンロード過程で使用される秘密鍵Ｋｓは、ダウンロードプログラムのデータ領域に暗号化されて格納されている。暗号化された秘密鍵Ｋｓの復号に必要な鍵は、定数としてダウンロードプログラムのコード中に分散して埋め込まれている。ダウンロードプログラムが証明の過程で使用する秘密鍵Ｋｓは、マイクロプロセッサ９０１が固有に有する秘密鍵Ｋｓと同じ値ではあるが、マイクロプロセッサ９０１から読み出して得られるのではない。マイクロプロセッサが固有に有する秘密鍵Ｋｓがプログラムによって読み出されることは決してないのである。

【００９４】
一般に、あるホストが特定の公開鍵を持つことを証明するには、認証局ＣＡ（Certificate Authority）が使われることが多い。本システムにＣＡを使う場合は、マイクロプロセッサ９０１の購入時に、マイクロプロセッサ９０１の公開鍵とユーザ（購入者）との組み合わせをＣＡに登録するか、オンラインでマクロプロセッサ９０１の公開鍵と利用者の組み合わせを登録する。この場合は、図１０に示すプログラム配布装置（以下、「サーバ」とする）９３により証明書の確認を省略することができる。

【００９５】
しかし、認証局ＣＡを利用するには、ユーザが購入したマイクロプロセッサ９０１を認証局に登録しなければならないという不便がある。そこで、第３実施形態では、サーバ９３とクライアント装置９１との間だけで認証が行えるように、暗号化された証明書をダウンロードプログラムに格納することとしている。なお、第３実施形態では、サーバ９３は証明書を盗用するなどの不正がいっさいない信用できるものであると仮定し、サーバ９３が全世界一意の鍵のペアＫ'pとＫ'sを持つことは、認証局などによってあらかじめ確認されているものとする。

【００９６】
図１０のシーケンスにおいて、まずシーケンス１００１において、クライアント装置９１とサーバ９３との間に、それぞれの装置の第１通信路設定部により、安全な第１の通信路を設定する。具体的には、クライアント装置９１から、ネットワーク９５を介してサーバ９３に通信開始要求を送り、通信路の秘密を守るための鍵共有を行う。これは周知の鍵共有プロトコル、ＤＨ方式などでよい。以後のクライアント装置９１とサーバ９３との間の通信は、こうしてネットワーク９５に設定された、盗聴に対して安全な通信路を通じて行う。

【００９７】
第１通信路が設定されたら、シーケンス１００２で、クライアント装置９１はサーバ９３に対してダウンロードしたいファイル（プログラム）を要求し、サーバ９３はクライアント装置９１とユーザレベルの認証や課金処理などを行う。ダウンロードの過程で、処理の秘密をユーザから守るため、以下に説明するクライアント装置９１でのダウンロードシーケンスの少なくとも一部は、暗号化コードとして実行される。ここでは、暗号化コードによって実行される部分を、マイクロプロセッサ９０１の動作として表現する。
【００９８】
シーケンス１００３では、第１通信路上に、クライアント装置９１のマイクロプロセッサ９０１とサーバ９３とを直接接続する、安全な第２の通信路を設定する。
【００９９】
本発明では、ユーザがダウンロード過程で処理されるデータの一部を不正に取得することを防止するため、ダウンロードプログラムでは、コード自身はもちろんのこと、処理過程でメモリ上に置かれるデータについても、ユーザが読み取り理解することが困難となるように記述されている。この目的をさらに強化するため、シーケンス１００３では、サーバ９３とマイクロプロセッサ９０１との間でも秘密鍵による通信の暗号化を行う。
【０１００】
秘密鍵の共有を行わない場合、ユーザによる不正、たとえば、サーバ９３とマイクロプロセッサ９０１との間の通信メッセージで、マイクロプロセッサ９０１の持つ公開鍵Ｋpを偽の公開鍵にすりかえて暗号化されたプログラムを取得し、偽の公開鍵に対応する既知の秘密鍵で復号化することによって平文のプログラムがユーザの手にわたるおそれがあるからである。サーバ９３とマイクロプロセッサ９０１との間の通信を秘密鍵で暗号化することにより、上述のようなユーザの不正を防ぐことができる。以後、「マイクロプロセッサ９０１とサーバ９３との間の通信」という場合、マイクロプロセッサ９０１上の耐タンパなプログラムとサーバ９３とが共有する暗号鍵によって暗号化され、保護された通信を意味する。
【０１０１】
安全な第２通信路の設定後、マイクロプロセッサ９０１とサーバ９３は相互認証を行う。すなわち、シーケンス１００４で、マイクロプロセッサ９０１はサーバ９３に対するチャレンジのための乱数Ｒｃを生成し、マイクロプロセッサ９０１に固有の公開鍵Ｋpとともに、第２通信路を介してサーバ９３に送信する。チャレンジを受け取ったサーバ９３は、シーケンス１００５で、サーバ９３の秘密鍵Ｋ'sで乱数Ｒｃを暗号化した署名$S_{K's}$［Ｒｃ]を、サーバ９３からのチャレンジＲｓおよびサーバ側公開鍵Ｋ'pとともに、マイクロプロセッサ９０１に送信する。（図１０では、署名$S_{K's}$［Ｒｃ]をＳ［Ｒｃ］(Ｋ's)と表記し、類似の表現も同様の表記とする。）
マイクロプロセッサ９０１は、サーバ９３から送られてきた署名$S_{K's}$［Ｒｃ]が、サーバの公開鍵Ｋ'ｐでハッシュ化された$V_{K'p}$［Ｒｃ]と一致するかどうかを確認する。不一致の場合は、サーバ９３の認証に失敗して、以後の処理を中止する。認証に成功すれば、シーケンス１００６でサーバ９３のチャレンジに対してS_{Ks}［Ｒｓ]を計算し、証明書Certをサーバ９３に送る。サーバ９３は、レスポンスS_{Ks}［Ｒｓ]をV_{Kp}［Ｒｓ]と比較して、一致しなければ処理を中止する。一致した場合は、マイクロプロセッサ９０１の公開鍵Ｋｐと認証機関の公開鍵Ｋvalとから、V_{Kval}［Ｋｐ]を計算し、証明書Certから公開鍵に対する署名S_{Kcert}［Ｋｐ]と照合し、検証に失敗すれば処理を中止する。
【０１０２】
照合できた場合は、$E_{K'p}$［Cert]を復号化してCertを取り出し、検証する。ここでも検証に失敗すれば処理を中止する。検証に成功して、証明書によりマイクロプロセッサ９０１が公開鍵Ｋｐを持つことを確認すると、サーバ９３は実行プログラムをマイクロプロセッサ９０１の公開鍵Ｋｐで暗号化した暗号化プログラムE_{Kp}［Ｐｒｏｇ］を作成する。このとき、第１実施形態で説明したように、プログラムのコード部分をマイクロプロセ

ッサ９０１の公開鍵Ｋｐで暗号化する。このとき、第１実施形態で説明したように、プログラムのコード部分をマイクロプロセッサ９０１の公開鍵Ｋｐで暗号化する。暗号化にあたっては、第１実施形態で説明したように、プログラム本体の.textセクションは暗号化するが、ジャンプテーブルの.IATセクションは平文のままとしておく。

【０１０３】

シーケンス１００７で、サーバ９３は暗号化プログラムE_{Kp}［Ｐｒｏｇ］と、サーバ９３自身の秘密鍵K'sによる署名$S_{K's}$［E_{Kp}[Prog]]とを、第２通信路を介してマイクロプロセッサ９０１に送る。この暗号化プログラムと署名は、直接マイクロプロセッサ９０１とサーバ９１との間に確立された第２通信路を介して送られるため、クライアント装置９１はこれを傍受することはできない。

【０１０４】

マイクロプロセッサ９０１はプログラムの受信を完了すると、シーケンス１００８でクライアント装置９１にダウンロード終了を通知する。シーケンス１００９で、クライアント装置９１はサーバ９３に課金処理を要求し、サーバ９３は領収書Rcptと、領収書のサーバ９３の秘密鍵による署名$S_{K's}$［Rcpt]およびプログラムの署名$S_{K's}$［E_{Kp}[Prog]]とをクライアント装置９１に送信する。クライアント装置９１は受信した領収書と署名を保存するとともに、シーケンス１０１０で、サーバ９３によるプログラムの署名$S_{K's}$［E_{Kp}[Prog]]をマイクロプロセッサ９０１に送る。

【０１０５】

マイクロプロセッサ９０１は、クライアント装置９１から受け取ったプログラムの書名$S_{K's}$［E_{Kp}[Prog]]をサーバ９３の公開鍵K'pで検証し、正当であれば、シーケンス１０１１で暗合された実行プログラムE_{Kp}[Prog]をクライアント装置９１に渡す。クライアント装置９１はE_{Kp}[Prog]を受け取ると、シーケンス１０１２でサーバ９３との間の回線の終了処理を行う。

【０１０６】

以後のマイクロプロセッサ９０１における暗号化プログラムの実行は、第１実施形態と同様である。

【０１０７】

図１１は、図１０におけるクライアント装置９１の処理フローをしめすフローチャートである。この処理フローは図１０のシーケンスで説明したとおりであるが、確認のため簡単に説明する。

【０１０８】

まず、ステップＳ１１０１でサーバ９３との間に第１の通信路を設定する。ステップＳ１１０３で、サーバ９３に対してプログラムのダウンロードの要求を行い、上述した認証処理を行う。ステップＳ１１０５でマイクロプロセッサ９０１とサーバ９３との間の認証を行い、プログラムをマイクロプロセッサ９０１に転送する。ステップＳ１１０７で、マイクロプロセッサ９０１から受信終了を受け取る。ステップＳ１１０９で、サーバ９３との間の課金処理を行い、ステップＳ１１１１で、サーバ９３から領収書と署名を受信する。ステップＳ１１１３で、マイクロプロセッサ９０１に領収書と署名を渡し、ステップＳ１１５で、マイクロプロセッサ９０１から暗号化されたプログラムを受け取る。ステップＳ１１１７で、サーバ９３との間の回線を終了する。

【０１０９】

図１２は、図１０におけるサーバ９３の処理フローを示すフローチャートである。まず、ステップＳ１２０１で、クライアント装置９１からの要求による第１通信路を設定する。ステップＳ１２０３で、クライアント装置９１からのプログラムのダウンロード要求を受信し、課金などの目的でクライアントとの認証を行う。ステップＳ１２０５で、マイクロプロセッサ９０１との間に第２の通信路を設定する。ステップＳ１２０７で、マイクロプロセッサ９０１からのチャレンジＲｃと公開鍵Ｋｐを受信する。ステップＳ１２０９で、サーバ側のチャレンジＲｓとレスポンスＳ［Ｒｃ］を生成し、サーバ９３の公開鍵K'pとともにＲｃ、Ｓ［Ｒｃ］をマイクロプロセッサ９０１に送信する。ステップＳ１２１１で

、マイクロプロセッサ９０１からのレスポンスＳ［Ｒｃ］と、証明書Ｅ[Cert]を受信する。ステップＳ１２１３で、マイクロプロセッサ９０１から受信したＳ［Ｒｃ］とＥ[Cert]が正当かどうか検証する。検証に失敗した場合は、処理を中止する。検証に成功した場合は、ステップＳ１２１５でマイクロプロセッサ９０１に対応した暗号化プログラムと署名を生成して、それらをマイクロプロセッサ９０１に送信する。ステップＳ１２１７で、マイクロプロセッサ９０１との間のプログラム転送を終了する。ステップＳ１２１９で、クライアント装置９１との間で課金処理を行い、領収書、署名を送信する。ステップＳ１２２１で、クライアント装置９１との間の回線を終了する。

【０１１０】

図１３は、図１０におけるマイクロプロセッサ９０１の処理フローを示すフローチャートである。まず、ステップＳ１３０１で、サーバ９３との間に第２の通信路を設定する。ステップＳ１３０３で、チャレンジＲｃを生成し、マイクロプロセッサ９０１に固有の公開鍵Ｋｐとともに、サーバ９３に送信する。送信が成功したなら、ステップＳ１３０５で、サーバ９３からレスポンスＳ［Ｒｃ］と、チャレンジＲｓおよび公開鍵Ｋ'pを受信する。ステップＳ１３０７で、レスポンスが正当かどうかを判断する。正当でないとされた場合は、処理を中止する。正当な場合は、ステップＳ１３０９で、サーバ９３に対するレスポンスと、サーバに対応した証明書を生成し、送信する。ステップＳ１３１１で、サーバ９３から暗号化されたプログラムを受信する。受信が終了すると、ステップＳ１３１３で、クライアント装置９１にダウンロードの終了を通知する。ステップＳ１３１５で、クライアント装置９１から署名を受信し、ステップＳ１３１７で、署名を検証する。検証が失敗すれば処理は中止する。検証に成功すれば、ステップＳ１３１９で、暗号化されたプログラムをクライアント装置９１に転送する。

【０１１１】

図１０のシーケンスにおいて、課金処理が完了した後のダウンロードプログラムの異常終了によって、暗号化されたプログラムＥ[Prog]が入手できなかった場合は、クライアントは保存した領収書をもとに、再度サーバ９３からプログラムをダウンロードする権利を保持する。この場合、当然課金は行われず、このマイクロプロセッサ９０１をターゲットとした暗号化プログラム以外のプログラムを得ることもない。

【０１１２】

また、図１０のプログラム配布シーケンスにおいて、サーバ９３が、プログラムの暗号化に、マイクロプロセッサ９０１の公開鍵を使用せず、共通鍵を使用してもよい。この場合は、サーバ９３が共通鍵Ｋｘと暗号化アルゴリズムを選択し、共通鍵Ｋｘでプログラムを暗号化した上で、この共通鍵をプロセッサに固有の公開鍵で暗号化し、サーバ９３とマイクロプロセッサ９０１との間の暗号化を行ってから、プログラムをマイクロプロセッサ９０１に送信する。このとき、第２通信路設定後に、マイクロプロセッサ９０１に備えられている暗号復号化能力をサーバ側からマイクロプロセッサ９０１に問い合わせるシーケンスを追加して、マイクロプロセッサ９０１に処理可能な暗号化アルゴリズムの中からサーバ９３が暗号化アルゴリズムを選択してもよい。

【０１１３】

実行コードの復号化にプログラムごとに使い捨ての共通鍵を使うので、鍵の長さを短くし、クライアント装置９１のメモリあるいはマイクロプロセッサ９０１のキャッシュの中に構築されるページテーブルのサイズを小さくできる。

【０１１４】

上述したようなダウンロードの手順は、データベースや顧客情報などの秘密情報の取り扱いにも適用できる。また、プログラムをターゲットのマイクロプロセッサだけで実行できるように暗号化することを除けば、実行プログラム以外の任意の音楽、映像データにも拡張して適用可能である。

【０１１５】

【発明の効果】

本発明のマイクロプロセッサによれば、マイクロプロセッサが組み込まれたＰＣの所有者

に解析されることなく、暗号化されたプログラムを安全に実行することができる。また、既存のプログラムにソース変更や再コンパイルすることなく、完全にプロセッサ内部で復号化を行うことができる。

【０１１６】

本発明の別の形態のマイクロプロセッサによれば、プログラムの暗号化に共通鍵を用い、共通鍵自体をマイクロプロセッサの公開鍵で暗号化した。これにより暗号化プログラムのセキュリティが補償されるとともに、マイクロプロセッサのハードウエアサイズを大幅に縮小することができる。

【０１１７】

本発明のプログラム配布システムでは、ネットワークを介して、プログラム配布装置からクライアント装置へ、安全かつ確実に暗号化プログラムを配布することができる。また、マイクロプロセッサで実行される耐タンパなダウンロードプログラムの存在を前提とすることにより、第三者を介さず、直接プログラム配布装置とマイクロプロセッサとの間で、プログラムのダウンロードを安全かつ効率良く行うことができる。

【０１１８】

また、コンピュータ読み取り可能な記録媒体に、プログラム本体部分を暗号化して記録し、外部のプログラムへの直接の参照を行うＩＡＴ領域は暗号化せずに平文の状態で格納することにより、プログラム実行時のリロケーションが正しく行われることを可能にする。

【図面の簡単な説明】

【図１】本発明の第１実施形態に係るマイクロプロセッサの概略ブロック図である。

【図２】本発明の第１実施形態に係るマイクロプロセッサで実行される暗号化プログラムのファイル形式の一例を示す図である。

【図３】本発明の第１実施形態に係るマイクロプロセッサの物理アドレスと、仮想記憶を管理するページテーブルと、論理アドレスとの対応関係を示す図である。

【図４】図３に示すページテーブルの中の、ページエントリの一例を示す図である。

【図５】本発明の第２実施形態に係るマイクロプロセッサの概略ブロック図である。

【図６】本発明の第２実施形態に係るマイクロプロセッサで実行される暗号化プログラムのファイル形式の一例を示す図である。

【図７】本発明の第２実施形態に係るマイクロプロセッサの物理アドレスと、仮想記憶を管理するページテーブルと、キーテーブルとの対応関係を示す図である。

【図８】図７に示すページテーブルとキーテーブルとの関係を詳細に示す図である。

【図９】本発明の第３実施形態に係る、プログラム配布システムの図である。

【図１０】本発明の第３実施形態に係るプログラム配布のシーケンス図である。

【図１１】図９に示すクライアント装置のプログラム受け取り手順を示すフローチャートである。

【図１２】図９に示すプログラム配布装置のプログラム配布手順を示すフローチャートである。

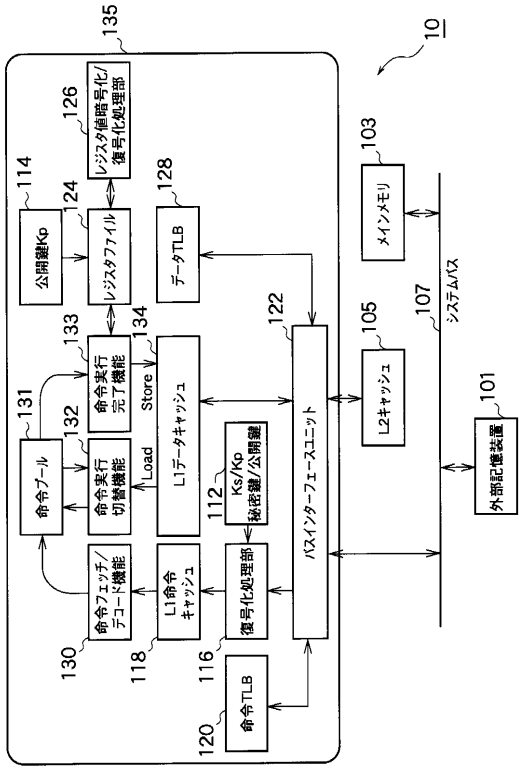
【図１３】図９に示すクライアント装置内部のマイクロプロセッサの動作を示すフローチャートである。

【符号の説明】

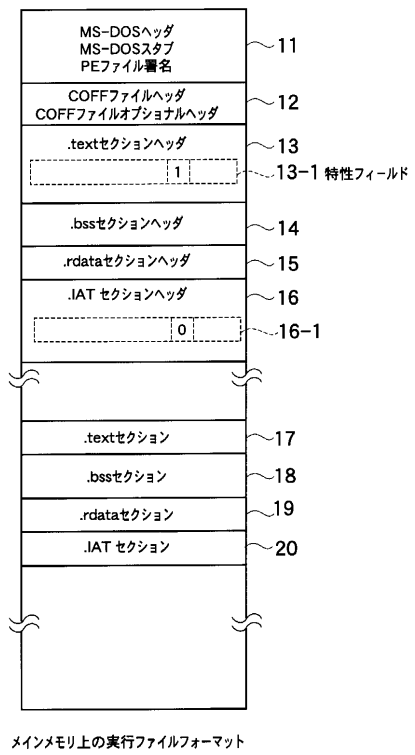
１０、２０、９０１　マイクロプロセッサ
３７、７７　ページテーブル
７９　キーテーブル
９１　クライアント装置
９３　プログラム配布装置
９５　ネットワーク
１０３、２０３　メインメモリ
１０５、２０５　二次キャッシュ
１１２、２１２　秘密鍵
１１４、２１４　公開鍵

１１６、２１６　復号化処理部
１１８、２１８　一次命令キャッシュ（第２の記憶手段）
１２０、２２０　命令ＴＬＢ（第１の記憶手段）
１２２、２２２　バスインターフェイスユニット
１２４、２２４　レジスタ
１２６、２２６　レジスタ値暗号化／復号化処理部
２３６　復号化鍵ＴＬＢ
９０５、９３２　第１通信設定部
９０６、９３４　第２通信設定部
９１０、９３３　ユーザ認証部
９０８、９１４　プログラム受信部
９３５　プログラム認証部
９３６　プログラム暗号化処理部

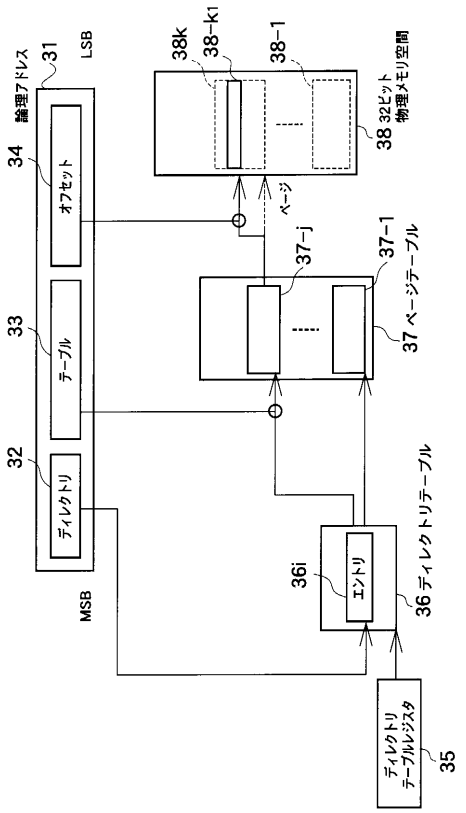
【図１】



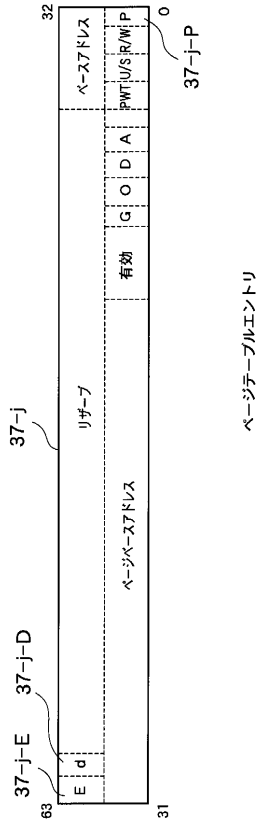
【図２】



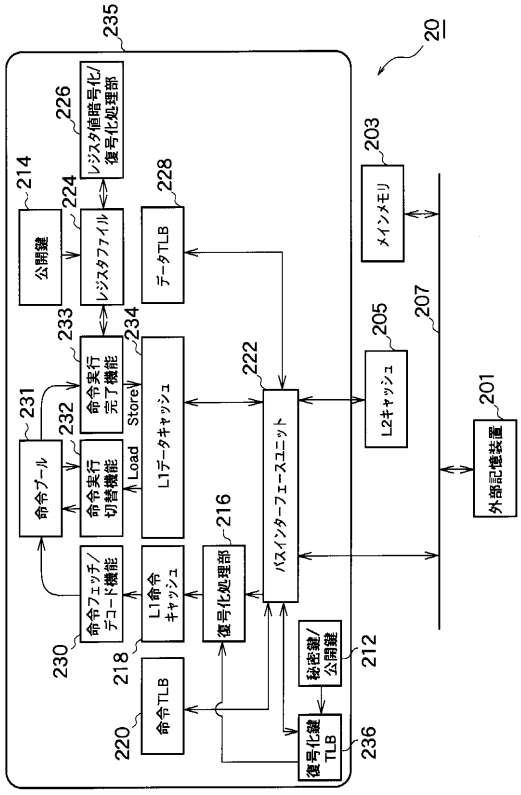
【図３】



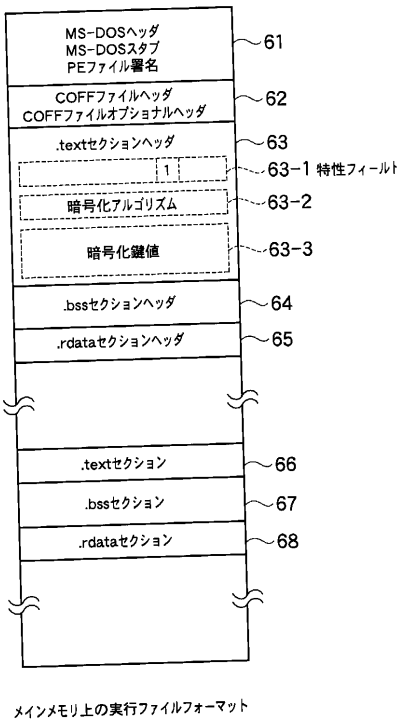
【図４】



【図５】

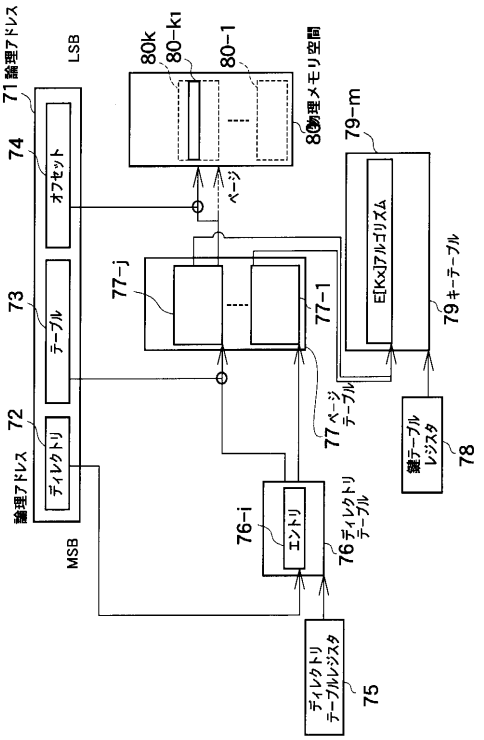


【図６】

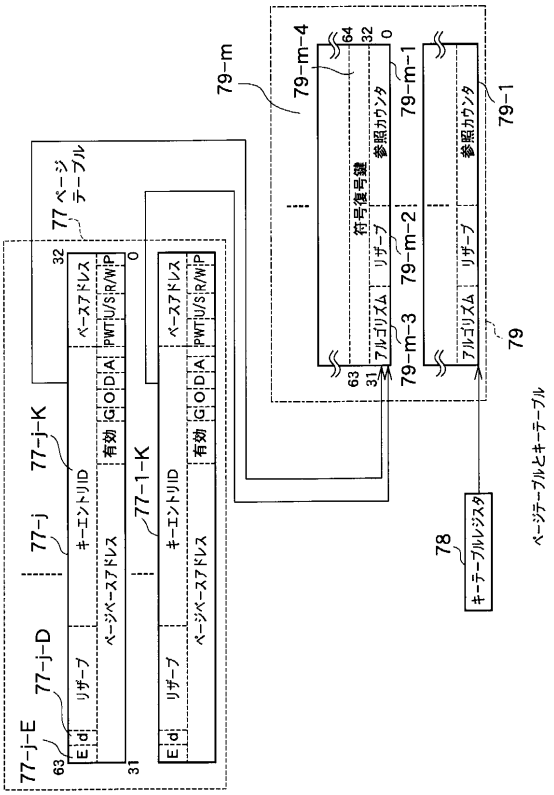


メインメモリ上の実行ファイルフォーマット

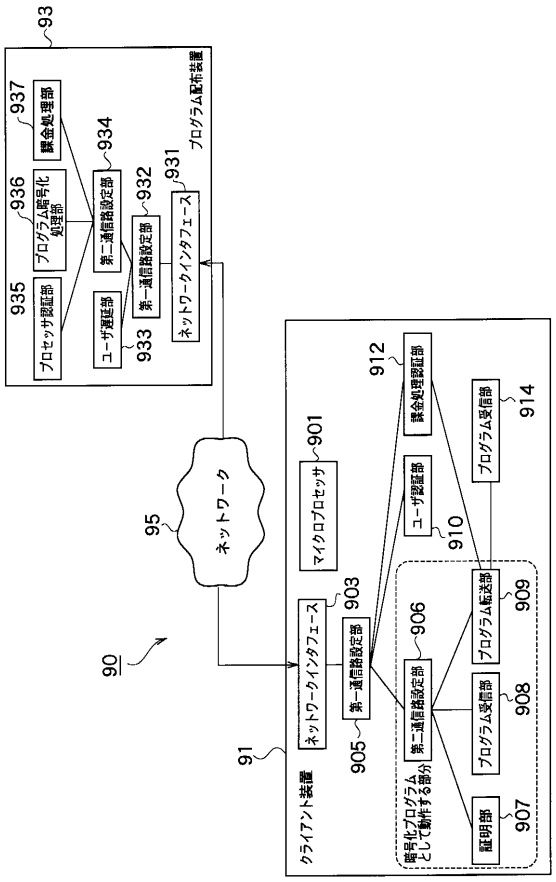
【図７】


71論理アドレス
LSB
MSB
74
オフセット
73
テーブル
72
ディレクトリ
論理アドレス
80k
80-k1
80-1
80物理メモリ空間
ページ
79-m
E[Kx]アルゴリズム
79キーテーブル
77-j
77-1
77ページテーブル
鍵テーブルレジスタ
78
76-i
エントリ
76ディレクトリテーブル
ディレクトリテーブルレジスタ
75

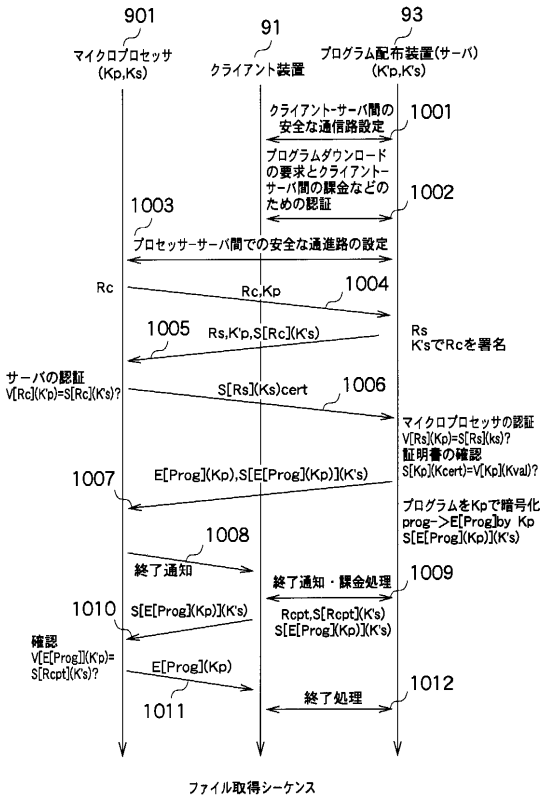
【図８】


79-m
79-m-4
79-m-1
79-1
64
32
0
63
31
符号復号鍵
リザーブ
参照カウンタ
アルゴリズム
79-m-3
79-m-2
79
77ページテーブル
77-j-K
77-j
77-j-D
77-j-E
77-1-K
ページベースアドレス
キーエントリID
有効
G
O
D
A
PWT
U/S
R/W
P
リザーブ
E
d
78
キーテーブルレジスタ
ページテーブルとキーテーブル

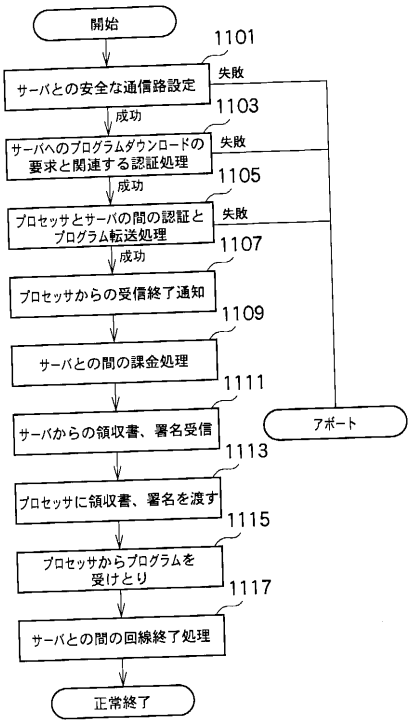
【図９】


93
プログラム配布装置
937
課金処理部
936
プログラム暗号化処理部
935
プロセッサ認証部
934
第二通信路設定部
932
第一通信路設定部
933
ユーザ認証部
931
ネットワークインタフェース
95
ネットワーク
90
91
クライアント装置
901
マイクロプロセッサ
903
第一通信路設定部
905
ネットワークインタフェース
912
課金処理認証部
910
ユーザ認証部
914
プログラム受信部
906
第二通信路設定部
暗号化プログラムとして動作する部分
909
プログラム転送部
908
プログラム受信部
907
証明部

【図１０】

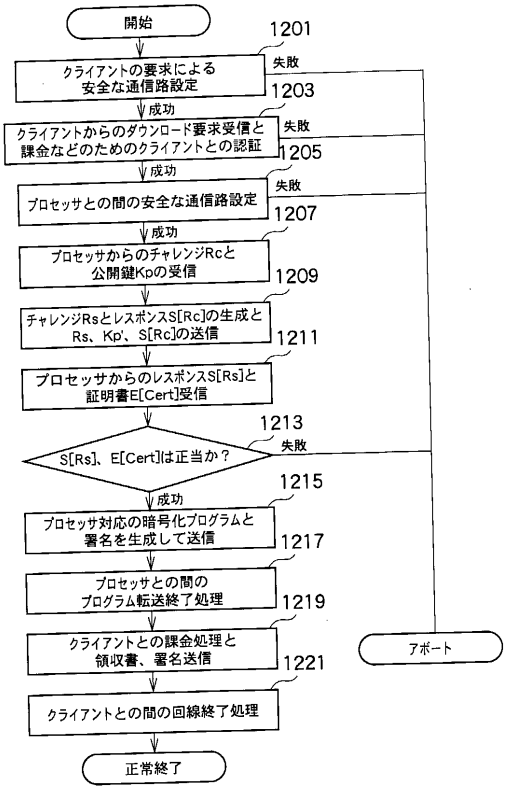

901
マイクロプロセッサ
(Kp,Ks)
91
クライアント装置
93
プログラム配布装置(サーバ)
(K'p,K's)
クライアント-サーバ間の安全な通信路設定
1001
プログラムダウンロードの要求とクライアント-サーバ間の課金などのための認証
1002
1003
プロセッサ-サーバ間での安全な通進路の設定
Rc
Rc,Kp
1004
1005
Rs,K'p,S[Rc](K's)
Rs
K'sでRcを署名
サーバの認証
V[Rc](K'p)=S[Rc](K's)?
S[Rs](Ks)cert
1006
マイクロプロセッサの認証
V[Rs](Kp)=S[Rs](Ks)?
証明書の確認
S[Kp](Kcert)=V[Kp](Kval)?
1007
E[Prog](Kp),S[E[Prog](Kp)](K's)
プログラムをKpで暗号化
prog->E[Prog]by Kp
S[E[Prog](Kp)](K's)
1008
終了通知
終了通知・課金処理
1009
1010
S[E[Prog](Kp)](K's)
Rcpt,S[Rcpt](K's)
S[E[Prog](Kp)](K's)
確認
V[E[Prog](Kp)]=
S[Rcpt](K's)?
E[Prog](Kp)
1011
終了処理
1012
ファイル取得シーケンス

【図１１】



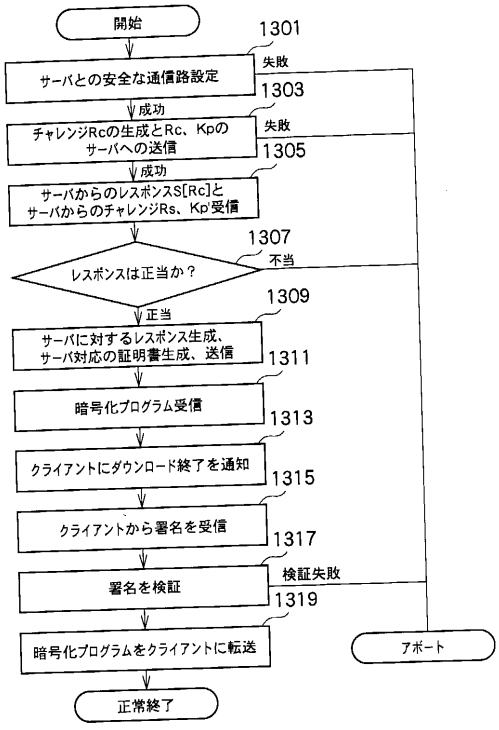
クライアントのフローチャート

【図１２】



サーバのファイル提供フローチャート

【図１３】



プロセッサのフローチャート

フロントページの続き

(74)代理人　100098327
　　弁理士　高松　俊雄
(72)発明者　橋本　幹生
　　神奈川県川崎市幸区小向東芝町１　株式会社東芝　研究開発センター内
(72)発明者　斉藤　健
　　神奈川県川崎市幸区小向東芝町１　株式会社東芝　研究開発センター内
(72)発明者　寺本　圭一
　　神奈川県川崎市幸区小向東芝町１　株式会社東芝　研究開発センター内

審査官　石田　信行

(56)参考文献　特開平０６－１１２９３７（ＪＰ，Ａ）
　　特開平０５－０２０１９７（ＪＰ，Ａ）
　　特開平０８－３０５５５８（ＪＰ，Ａ）
　　特開平１１－２８２６６７（ＪＰ，Ａ）
　　特表２００１－５２７６７５（ＪＰ，Ａ）
　　特開２００１－０４３１３９（ＪＰ，Ａ）
　　特開平１１－２０３１２８（ＪＰ，Ａ）
　　特表２００２－５２６８２２（ＪＰ，Ａ）
　　特開２００２－１４０２３６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
　　H04L　9/10
　　G09C　1/00
　　G06F 12/14