(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4375980号
(P4375980)

(45)発行日　平成21年12月2日(2009.12.2)　(24)登録日　平成21年9月18日(2009.9.18)

(51)Int.Cl.		F I		
G06F 12/14	(2006.01)	G06F	12/14	510E
G06F 21/24	(2006.01)	G06F	12/14	520A
G06F 9/48	(2006.01)	G06F	9/46	452Z
H04L 9/08	(2006.01)	H04L	9/00	601A

請求項の数 13　(全 20 頁)

(21)出願番号　特願2003-65685(P2003-65685)
(22)出願日　平成15年3月11日(2003.3.11)
(65)公開番号　特開2004-272816(P2004-272816A)
(43)公開日　平成16年9月30日(2004.9.30)
審査請求日　平成17年4月11日(2005.4.11)

前置審査

(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　稲村　雄
東京都千代田区永田町二丁目１１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１１番１号
株式会社エヌ・ティ・ティ・ドコモ内

審査官　平井　誠

最終頁に続く

(54)【発明の名称】マルチタスク実行システム及びマルチタスク実行方法

(57)【特許請求の範囲】
【請求項１】
単一のＣＰＵによって第１タスクおよび第２タスクを含む複数のタスクを順次実行し、前記複数のタスクを仮想的に並列処理するマルチタスク実行システムであって、
前記第１タスクに対応付けられ、前記第１タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第１機密情報を格納する第１機密情報保持領域と、
前記第２タスクに対応付けられ、前記第２タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第２機密情報を格納する第２機密情報保持領域と、
前記第１機密情報の暗号化に用いられる第１鍵情報と、前記第２機密情報の暗号化に用いられる第２鍵情報とを格納する暗号化鍵記憶領域と、
前記ＣＰＵが実行するタスクをオペレーティングシステムによって前記第１タスクから前記第２タスクへ切り替えるタスク切り替えの際に、停止状態となる前記第１タスクの前記第１機密情報を前記第１鍵情報を用いて暗号化する暗号処理部と
を備え、
前記第１機密情報保持領域は、仮想メモリ内に設けられ、
前記マルチタスク実行システムは、前記第１機密情報がスワップファイル等の二次記憶領域に保存されたか否かを判定する退避判定手段をさらに備え、
前記暗号処理部は、前記保存が実行されていない場合に、前記第１機密情報に対する暗号化を実行することを特徴とするマルチタスク実行システム。
【請求項２】

複数のＣＰＵによって第１タスクおよび第２タスクを含む複数のタスクを順次実行し、前記複数のタスクを仮想的に並列処理するマルチタスク実行システムであって、
前記第１タスクに対応付けられ、前記第１タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第１機密情報を格納する第１機密情報保持領域と、
前記第２タスクに対応付けられ、前記第２タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第２機密情報を格納する第２機密情報保持領域と、
前記第１機密情報の暗号化に用いられる第１鍵情報と、前記第２機密情報の暗号化に用いられる第２鍵情報とを格納する暗号化鍵記憶領域と、
前記ＣＰＵが実行するタスクをオペレーティングシステムによって前記第１タスクから前記第２タスクへ切り替えるタスク切り替えの際に、停止状態となる前記第１タスクの前記第１機密情報を前記第１鍵情報を用いて暗号化する暗号処理部と、
前記第１機密情報保持領域および前記第２機密情報保持領域にアクセス権を設定し、前記第１機密情報保持領域および前記第２機密情報保持領域に対するアクセス制限を行うアクセス制御部と
を備えるマルチタスク実行システム。
【請求項３】
前記暗号処理部は、前記ＣＰＵとは別途独立に設けられたハードウェアであることを特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項４】
前記暗号処理部により暗号化された前記第１機密情報を、前記タスク切り替え時に、前記第１鍵情報を用いて復号する復号処理部を有することを特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項５】
前記復号処理部は、前記ＣＰＵとは別途独立に設けられたハードウェアであることを特徴とする請求項４に記載のマルチタスク実行システム。
【請求項６】
前記第１機密情報保持領域に格納される前記第１機密情報及び格納先の前記第１機密情報保持領域は、前記オペレーティングシステム上で実行されるアプリケーション内で指定されること特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項７】
前記第１機密情報保持領域は、仮想アドレスと物理アドレスとのマッピングによって確保され、前記マッピングに際し、前記暗号化鍵記憶領域に対するアクセス制限を行うマッピング管理部を備えることを特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項８】
前記第１鍵情報は、前記第１タスクの生成処理時に生成されるものであることを特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項９】
前記第１鍵情報は、前記オペレーティングシステムの起動時に生成されるものであることを特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項１０】
新規なタスクの生成処理時に、前記複数のタスクのいずれかである既存のタスクを複製して生成された新規なタスクと、前記既存のタスクとで共有される機密情報保持領域に格納される情報について、前記新規なタスクに対し、前記既存のタスクに対応付けられた鍵情報とは別途独立な鍵情報を生成することを特徴とする請求項１又は２に記載のマルチタスク実行システム。
【請求項１１】
単一のＣＰＵによって第１タスクおよび第２タスクを含む複数のタスクを順次実行し、前記複数のタスクを仮想的に並列処理するマルチタスク実行方法であって、
前記第１タスクに対応付けられ、前記第１タスクが処理する情報の少なくとも一部であ

って他のタスクから機密にすべき第１機密情報を格納する第１機密情報保持領域を確保するステップ（１）と、
前記第２タスクに対応付けられ、前記第２タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第２機密情報を格納する第２機密情報保持領域を確保するステップ（２）と、
前記第１機密情報の暗号化に用いられる第１鍵情報と、前記第２機密情報の暗号化に用いられる第２鍵情報とを生成するステップ（３）と、
前記ＣＰＵが実行するタスクをオペレーティングシステムによって前記第１タスクから前記第２タスクへ切り替えるタスク切り替えの際に、停止状態となる前記第１タスクの前記第１機密情報を前記第１鍵情報を用いて暗号化するステップ（４）と
を備え、
前記第１機密情報保持領域は、仮想メモリ内に設けられ、
前記ステップ（４）では、前記第１機密情報がスワップファイル等の二次記憶領域に保存されていない場合に、前記第１機密情報に対する暗号化を実行することを特徴とするマルチタスク実行方法。
【請求項１２】
複数のＣＰＵによって第１タスクおよび第２タスクを含む複数のタスクを順次実行し、前記複数のタスクを仮想的に並列処理するマルチタスク実行方法であって、
前記第１タスクに対応付けられ、前記第１タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第１機密情報を格納する第１機密情報保持領域を確保するステップ（１）と、
前記第２タスクに対応付けられ、前記第２タスクが処理する情報の少なくとも一部であって他のタスクから機密にすべき第２機密情報を格納する第２機密情報保持領域を確保するステップ（２）と、
前記第１機密情報の暗号化に用いられる第１鍵情報と、前記第２機密情報の暗号化に用いられる第２鍵情報とを生成するステップ（３）と、
前記ＣＰＵが実行するタスクをオペレーティングシステムによって前記第１タスクから前記第２タスクへ切り替えるタスク切り替えの際に、停止状態となる前記第１タスクの前記第１機密情報を前記第１鍵情報を用いて暗号化するステップ（４）と、
前記第１機密情報保持領域および前記第２機密情報保持領域にアクセス権を設定し、前記第１機密情報保持領域および前記第２機密情報保持領域に対するアクセス制限を行うステップ（４）と
を備えることを特徴とするマルチタスク実行方法。
【請求項１３】
暗号化された前記第１機密情報を、前記タスク切り替え時に、前記第１鍵情報を用いて復号するステップをさらに有することを特徴とする請求項１１又は１２に記載のマルチタスク実行方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、コンピュータにおけるセキュリティ技術に係り、擬似的(TSS環境)もしくは実際に(ＳＭＰ環境)複数のタスクを同時並列に実行することができるマルチタスク実行システム及びマルチタスク実行方法に関する。
【０００２】
【従来の技術】
近年、個人の使用するパーソナルコンピュータ(PC)から電子商取引(EC)等のサービスを提供するサーバコンピュータ、さらにはスーパーコンピュータまで、事実上あらゆる計算機がマルチタスク環境を提供するものとなっている。
【０００３】
ここでいうマルチタスク環境とは、計算機が扱う処理の単位である『タスク』を複数個同

時に実行させることを可能とする環境であり、それらのタスクは、単一の中央演算装置(ＣＰＵ)を用いた計算機では、TSS(Time Sharing System,時分割型)による擬似的な並列実行を、複数のＣＰＵを実装した計算機ではＳＭＰ(Symmetric Multi Processor)構成を取ることで、真に同時並列的な実行を実現している。

【０００４】

このようなマルチタスク環境では、どのようにして個々のタスクが保持する機密データを他のタスクから保護するのかが問題となる。これは電子商取引に用いるサーバなど重要な情報を扱わなければならない計算機では特に影響が大きいと考えられる。

【０００５】

例えば、インターネットを利用して電子商取引を行う場合、SSL（Secure Sockets Layer、米国特許第5,825,890号。Elgamal, et al. "Secure socket layer application program apparatus and method",0ctober 20,1998）と呼ばれる仕組みでブラウザ／ウェブサーバ間の通信を保護するというのが一般的となっている。

【０００６】

ところが、このSSLを安全に機能させるためには、ウェブサーバに当該サーバ以外には知ることのできない固有の秘密鍵をインストールする必要がある。サーバはアクセスしたユーザに対してその秘密鍵を利用して自らの身元を証明するが、万一この秘密鍵が他者の知るところになれば、当該他者は

1.当該秘密鍵を利用して当該サーバへのなりすまし

2.当該秘密鍵に対応した公開鍵によって暗号化されるブラウザ／サーバの通信セッション開始時のやり取りを傍受することにより、そこで交換された共通鍵によって暗号化される実際の通信の復号及び内容の傍受

といった攻撃が可能となってしまう。

【０００７】

このような危険を避けるため、従来では、当該秘密鍵は一般的には暗号化処理を受けた上でファイル等に保管されるような運用がなされている。これにより、ディスク上に保管されている状態では秘密鍵の安全性は保護のために利用される暗号技術の強度の範囲内で保証されるが、ウェブサーバプログラムが実行されるときには、当該秘密鍵はファイルから読み出され、復号された上で当該アプリケーションプログラムのために確保されたメモリ上に平文（非暗号文）のままで保存されることになる。

【０００８】

このとき、同じサーバ計算機上で同時に実行される他のタスクが当該秘密鍵情報の保存されているメモリ領域にアクセスし、平文状態のデータを読み出すことができれば、容易にウェブサーバの秘密鍵を知ることができてしまうため、上に述べたような攻撃の実施が可能となる。このように、機密情報を保持するメモリを同時実行される他のタスクから保護することは、マルチタスクが一般的となっている現代的な計算機環境では特に重要性を増している。

【０００９】

マルチタスク環境をサポートする多くのオペレーティングシステムでは、タスク毎に仮想記憶空間を割り当て、各タスクがアクセスする仮想的なメモリ領域と物理的なメモリ領域とのマッピングを管理することにより、一つのタスクが保持するメモリ上のデータに対して他のタスクがアクセスすることはできないような配慮がなされている。

【００１０】

しかしながら、実際には、稼動中のタスクに関してデバッグが必要な場合等のために実行中のタスクに対してデバッガを接続し、当該タスクのメモリ領域や実行状態をモニタリングする機能が提供されているため、このモニタリング機能を悪用した精査、いわゆるハッキングが可能となることから、上述したようなオペレーティングシステムによる保護は完全ではない。特に悪意を持った計算機の管理者権限を持つユーザが存在する場合、上記モニタリング機能を利用して、秘匿情報の読み出しが可能となる。

【００１１】

なお、ここでいう『管理者権限を持つユーザ』は、必ずしも正規の計算機管理者であるとは限らない。近年の計算機環境に対して大きな脅威となっているのは、管理者権限下で実行することが要請されるタスクのプログラムコード中に存在するセキュリティ・ホールを利用して、"シェル"と呼ばれるコマンド実行環境を奪取するというものであり、管理者権限の下で動作するそのようなシェルを入手した攻撃者は、管理者として事実上任意の操作を行えることになる。そのような操作としては、先に述べたデバッガによる実行中タスクの精査の他に

・タスクのメモリ空間に存在する全情報のファイルへの書き出し

・実メモリ容量の枯渇に伴いスワップファイルと呼ばれるハードディスク等の二次記憶領域に待避されたタスクメモリの精査

等の攻撃が考えられる。

【００１２】

また、前述したSSLを始めとして暗号技術を利用した通信データの保護を提供する仕組みは各種考案されているが、それらは事実上すべて秘密鍵をメモリに読みこんだ形でタスクが実行されるようになっているため、ネットワーク上ではなくその実行環境である計算機自体に対して攻撃が加えられる場合、いずれも大きな脆弱性を持つことが知られている。

【００１３】

このような問題を解決するための従来技術としては、大別して二種類の解決策が考案されいる。一つは管理者権限の細分化という主としてソフトウェア的な解決策であり、もう一つは機密情報を物理的／論理的に隔離するという手立てである。

【００１４】

前者に属する技術としては、いわゆる"トラステッドＯＳ"として知られるオペレーティングシステムが該当する。これらのオペレーティングシステムは、基本的には管理者権限を細分化することで絶対的な権限を持ったタスクの実行を可能な限り減らすという戦略を採っている。また、アクセスコントロールリストと呼ばれる機能を用いることで、ファイル等のデータに対するアクセス管理も従来のオペレーティングシステムより詳細に設定可能となっており、あるタスクが同時並列的に実行される他のタスクの情報にアクセスすることが非常に困難であるように図られている。

【００１５】

この種のトラステッドＯＳが採用している権限及びアクセス管理手法の詳細化による保護強化という方針は、基本的にはオペレーティングシステムというソフトウェアのみで実現できるため、特別なハードウェアを必要とせず比較的低コストで済むという利点がある。一方、オペレーティングシステムの保護機能によって他のタスクから隔離されているとはいえ、タスクのメモリ上ではデータは平文のまま保持されているため、同メモリ上のデータに攻撃者がアクセスする手段が開発された場合、機密情報が漏洩する可能性がある。

【００１６】

一方、後者の対策に関してはさらに

ａ）機密情報を扱う部分を計算機から物理的に分離することによる隔離

ｂ）機密情報を含むデータを単一のタスクのみが復号できる形で暗号化することによる保護

という二つの方向性がある。

【００１７】

ａ）はいわゆる"耐タンパー性デバイス"、すなわち、内部に保持されているデータを読み出すことが困難であるような機器に機密データを保持することで他者からの干渉を排除しようというものである。耐タンパー性デバイスは一般にデータの保存領域の他に独自の演算処理装置も装備されているため、機密情報を外部に洩らすことなく処理することを可能とする。

【００１８】

このような耐タンパー性デバイスによって機密情報の防御を行う例としては、個人の秘密情報を保護するためのスマート・カードや、公開鍵暗号技術を利用してデジタル証明書を

発行する認証機関の秘密鍵を保護するために主として使われるHSM（ハードウェア・セキュリティ・モジュール)等がある。また、暗号関連処理の高速化を主目的とするSSLアクセラレータ等のハードウェア製品のなかにも、秘密鍵等の機密情報を外から読み出すことができないような構成となっているものが存在するが、そのような製品もこのａ）に相当する対策とみなすことができる。

【００１９】

しかしながら、このような耐タンパー性デバイスを用いる対策には本質的に柔軟性に欠けるという欠点が存在する。大半の耐タンパー性デバイスは例えば『入力として与えられた暗号化済みデータを秘密鍵によって復号する』といった特定の機能しか持っていないため、当該デバイスがサポートしている以外の形で機密情報を保護することは困難となる。

【００２０】

耐タンパー性デバイスのなかにはJava（登録商標）アプレット等の形で任意のアプリケーションをインストールすることにより、機能を拡張できるようになっているものも存在するが、そのような形で柔軟性を得ようとすると、今度はインストールされるアプリケーション自体がウイルス等の悪意あるものではないことを保証することが難しくなる。

【００２１】

また、大半の機密情報というものはその値そのものを知られないことが重要である他、実際にはその機密情報を用いた処理を権限者以外に行われてしまうという状況も防がなければならない。攻撃者にとっては秘密鍵そのものを知ることとその秘密鍵を用いて任意のデータに対して電子署名をすることとはほとんど差異がないからである。耐タンパー性デバイスによって秘密情報そのものを他者に知られる可能性自体は低く抑えられるとしても、当該デバイスが権限を持たない者による指示に基づいて当該秘密情報を用いた操作(例えば秘密鍵による暗号化済みデータの復号処理)を行い、その結果を返してしまうときには、所期の目的は達成されないことになる。

【００２２】

一般に、計算機に接続される耐タンパー性デバイス等の外部デバイスへのアクセスはオペレーティングシステムにより管理されることになるが、そのアクセス管理機能が侵された場合、マルチタスク環境では権限を持たないユーザにより当該耐タンパー性デバイスを濫用される可能性が排除できない。

【００２３】

一方、上記ｂ）は、タスクのメモリ中に存在するデータを暗号化することによって他のタスクによる干渉から回避するものである。この方法では、メモリに保存されたデータをそのタスクでしか復号できないような形で暗号化しておくことにより、他のタスクが当該タスクのメモリにアクセスしたとしても、そのデータは解読できない暗号データとしてしか認識されないため、機密データに対してａ)の方策よりも強力な保護を提供することが可能となる。

【００２４】

例えば、従来技術である米国特許第5,224,166号(Hartman,Jr."System for seamless processing of encrypted and non-encrypted data and instructions",June 29,1993)では、外部記憶及びメモリ上ではデータ領域／コード領域ともに暗号化された形で保存され、それらのデータ／コードを中央演算装置内の物理的に安全な領域に置かれたキャッシュ領域に読み込む際に復号することで他のタスクからの不正なアクセスを排除するシステムが提案されている。

【００２５】

暗号化されたデータを復号するための鍵も同じく物理的に安全な領域中に置かれ、不正な利用が不可能となるように管理される。このシステムでは復号された平文データは物理的に安全な領域に位置するキャッシュ領域のみに置かれ、それらがＣＰＵのキャッシュ・ライトバックによりメモリに書き戻される際には自動的に暗号化されることで機密性が確保される。

【００２６】

また、別の従来技術である日本国特許、特開2001-318787号(橋本幹生他,"マイクロプロセッサ、これを用いたマルチタスク実行方法、及びマルチスレッド実行方法",平成12年5月8日出願)（特許文献２）では、同様にメモリ上のデータをキャッシュに読み出す、もしくは逆方向に書き出す際に復号処理／暗号化処理を行うことに加えて、コンテクスト切り替えと呼ばれるオペレーティングシステム機能実施時に実行レジスタ等のタスクの固有情報をも暗号化することで、前述したデバッガ利用等による攻撃をも未然に防ぐシステムが提案されている。

【００２７】

【特許文献１】

米国特許第5,224,166号明細書

【００２８】

【特許文献２】

特開2001-318787号公報

【００２９】

【発明が解決しようとする課題】

上述した、二例の従来技術は、基本的にはマイクロプロセッサに対する追加機能としてデータもしくはコードの暗号化／復号を行うためのハードウェアモジュールをプロセッサ内部(プロセッサ内キャッシュとシステムバスとの間)に実装するというものである。

【００３０】

これらの従来技術では、ハードウェア的にマイクロプロセッサを拡張するという手法は最大限のパフォーマンス的な利益を得ることをできるため、これら従来技術のようにメモリ中に置かれるデータ領域及びコード領域すべてを暗号化し、実行の過程でそれらをリアルタイムに復号／再暗号化するというようなシステムを実現するためには、最適なものと考えられる。

【００３１】

しかしながら、近年の最先端のマイクロプロセッサは高度な研究開発力及び技術的ノウハウの集積物であり、そのような既存のマイクロプロセッサシステムのなかに新たなハードウェアを追加することで全体的な複雑性を増大させることは実用的ではない。

【００３２】

また、すべてのデータをメモリからキャッシュへの読み出し、もしくは書き出しというタイミングで動的に復号／暗号化処理を行うという仕組みは、実行性能に関して大きな悪影響を与えることが考えられる。例えば、メモリアクセスに関して現在の最新テクノロジーでは数GB/secという最大性能が得られるが、ハードウェアによる暗号化処理性能は高々Gb/secレベルであるため、それだけでも純粋なメモリアクセス性能が1／10以下に落ちる可能性がある。

【００３３】

近年のマイクロプロセッサでは、基本的により高速なプロセッサをより低コストで提供する要請が大きいため、このような性能に対して負の影響を与えるような機能追加は妥当ではないといえる。このため、マルチタスク環境での機密保護という仕組みも、マイクロプロセッサ本体の機能とは分離して実現することが望まれる。

【００３４】

そこで、本発明は、以上の点に鑑みてなされたもので、マルチタスク環境をサポートする現代的な計算機オペレーティングシステムにおいて、同時並列的に実行される複数のタスクが他のタスクから重要な情報を読み出されることなしに実行できる環境を実現することのできるマルチタスク実行システム及びマルチタスク実行方法を提供することを目的とする。

【００３５】

【課題を解決するための手段】

上記課題を解決するために、本発明は、単一のＣＰＵによって複数のタスクを順次実行し、複数のタスクを仮想的に並列処理するマルチタスク実行システムであって、各タスク毎

に対応付けられる機密情報保持領域と、各機密情報保持領域に固有に設定され、各機密情報保持領域を暗号化するための鍵情報を格納する暗号化鍵記憶領域と、オペレーティングシステムによるタスク切り替え時に停止状態となるタスクに関連付けられた機密情報保持領域を、当該タスク切り替え時に、鍵情報により暗号化する暗号処理部とを備える。

【００３６】

また、本発明は、複数のＣＰＵにより複数のタスクを並列処理するマルチタスク実行システムであって、各タスク毎に対応付けられる機密情報保持領域と、各機密情報保持領域に固有に設定され、各機密情報保持領域を暗号化するための鍵情報を格納する暗号化鍵記憶領域と、オペレーティングシステムによるタスク切り替え時に停止状態となるタスクに関連付けられた機密情報保持領域を、当該タスク切り替え時に、鍵情報により暗号化する暗号処理部と、機密情報保持領域毎にアクセス権を設定し、機密情報保持領域に対するアクセス制限を行うアクセス制御部とを備える。

【００３７】

上記発明においては、暗号処理部により暗号化された機密情報保持領域を、当該タスク切り替え時に、前記鍵情報により復号する復号処理部を有することが好ましい。なお、上記暗号処理部及び復号処理部は、ＣＰＵとは別途独立に設けられたハードウェアとすることができる。

【００３８】

上記発明においては、鍵情報は、タスクの生成処理時や、オペレーティングシステムの起動時に生成することができる。この場合、新規なタスクの生成処理時に、既存のタスクから複製され、新規なタスクと既存のタスクとで共有される機密情報保持領域に格納される情報について、当該新規なタスクに対し、既存のタスクに対応付けられた鍵情報とは別途独立な鍵情報を生成することが好ましい。

【００３９】

このような上記発明では、タスクの切り替え、すなわちコンテクストスイッチを行うオペレーティングシステムのカーネルの動作に基づいて、コンテクストスイッチが実行されるタイミングで機密情報の暗号化及び復号処理を行う。詳述すると、あるタスクに割り当てられたＣＰＵ利用時間が終了した、或いは入出力待ち等で実行が中断する等の原因で発生するコンテクストスイッチ処理では、レジスタ上のデータも含めて当該タスクの実行に必要なすべての情報が当該タスク専用に割り当てられる領域に保存された上で、同様に保存されている別のタスクの情報をレジスタにロードし実行が再開される、という処理が行われる。

【００４０】

すなわち、本発明においてカーネルは、コンテクストスイッチに際し、

(1)実行状態から休止状態に移行させられるタスクに関しては当該タスクが保持する機密情報をそのタスク専用の鍵で暗号化する

(2)休止状態から実行状態に復帰させられるタスクに関しては、前回休止状態に移行される際に暗号化されている機密情報を当該タスク専用の鍵で復号する

といった処理を実行する。

【００４１】

また、本発明によれば、システムコールを利用するアプリケーションプログラムのプログラミング時において、機密性が必要とされるデータ及び当該データを格納する機密情報保持領域を明示的に指定し、その特定領域のみを、リアルタイムにではなく必要に応じた特定のタイミング、すなわちタスクの切り替え時に秘匿処理を行うことができる。なお、ここでいうデータの秘匿処理には、従来より利用されているような各種暗号技術利用したものの他、情報隠蔽技術(Steganography)を用いることができる。

【００４２】

さらに、本発明によれば、プログラム実行中にリアルタイムでの秘匿処理を実施する必要はなくなるため、マイクロプロセッサ内部に特別な機構を加える必要がなく、チップセットや拡張ボード等のハードウェアを追加するという形態や、さらにはソフトウェアのみに

よる形態で実現することが可能となり、既存のシステム及び装置を有効に利用することができる。

【００４３】

また、上記発明においては、タスクの仮想メモリ中の機密情報がスワップファイル等の二次記憶領域に保存されたか否かを判断する退避判定手段を備え、暗号処理部は、保存が実行されていない場合に、暗号化処理を実行することが好ましい。

【００４４】

この場合には、二次記憶上のスワップファイルに退避させられたデータを精査し、タスクのメモリ上に存在する重要な機密データがハッキングされる可能性を回避することができる。すなわち、複数のタスクが仮想的に同時並列的に実行されるマルチタスク実行環境において、一つのタスクが実行されている場合に、それ以外のすべてのタスクが停止状態となり、実行継続に必要となる情報を保存するシステムであっても、デバッガによる実行中のタスク内部状態を精査し、保存された情報を解析し、秘匿情報が漏洩することを回避することができる。

【００４５】

また、本発明では、スワップファイルへの書き出しの時点における暗号化処理を要することなく、退避が必要なデータを保護することができる。詳述すると、スワップファイルへのメモリデータの書き出しという処理が発生するのは、実行中のタスクからのメモリ要求が満たせなかった場合であり、この書き出し処理はカーネルの機能によって行われる。カーネルが動作する際には通常のユーザタスクはすべて休止状態にあるため、本発明を適用されたシステムの場合、それらタスクが保持する機密情報は既に暗号化された状態となっており、メモリ上に存在する任意のデータがスワップファイルに書き出され、さらにそのスワップファイルの中身が攻撃者によって精査されたとしても、機密情報の漏洩を回避することができる。。

【００４６】

さらに、本発明によれば、複数のＣＰＵを搭載するいわゆるＳＭＰ(Symmetric Multi Processor)構成の計算機において、同時に複数のタスクがそれぞれ別個のＣＰＵを占有することにより、真に同時並列的に実行が行うシステムであっても、あるタスクの実行中に、機密データ保持領域が平文のままという状態で存在するのを回避し、他のタスクからの操作により、当該領域が漏出するのを防止することができる。

【００４７】

すなわち、本発明では、他のタスクによって機密データが漏出するのを防止するために、ＳＭＰ構成の計算機で用いる場合には、単にコンテクストスイッチ時に暗号化／復号化を行うだけではなく、実行中の機密情報を保持するメモリ領域が他のタスクからアクセスできないように制限することができる。

【００４８】

具体的には、メモリ領域に対して特定のタスクのみからしか参照できないという属性の付与するとともに、機密情報保持用メモリ領域には同属性を付与する、といった機能をＳＭＰ対応オペレーティングシステムカーネルに追加する。

【００４９】

なお、メモリデータにも一般的なファイルと同じく書き込み権限、読み出し権限、実行権限等の各種属性が存在し、これらの各属性の設定をサポートするために通常オペレーティングシステムにはあるメモリ領域がどのような属性を持っているのかを管理する機能が備えられていることから、既存のオペレーティングシステムの構成を大幅に変更することなく、上記属性を追加することは可能である。

【００５０】

【発明の実施の形態】

［第１実施形態］

本発明の第１実施形態について説明する。図１は、本発明のマルチタスク実行システムを単一ＣＰＵシステムに適用した場合を例示するブロック図である。

【００５１】
同図に示すように、本実施形態の情報秘匿システムは、ハードウェア１００と、メモリ空間２００とから構成されている。ハードウェア１００は、マイクロプロセッサ１０１と、メモリ１０２と、暗号処理エンジン１０３とがシステムバス１０４によって結合されている構成される。
【００５２】
なお、暗号処理エンジン１０３は、システムバス１０４ではなくPCIのような、I／Oコントローラ１０５を介して、ハードウェア１００の外部に接続された外部バス１０７に実装される増設ボード１０６など、メモリ１０２との間でのDMA転送が可能な装置を採用することができる。また、暗号処理エンジン１０３としては、上述した１０３や１０６のように、ハードウェアを用いずに、オペレーティングシステム自身がソフトウェア的に暗号化処理を実施するようにしてもよい。すなわち、本実施形態に係るマルチタスク実行システムでは、特定のハードウェア形態に捕らわれずに情報秘匿を実施できるという柔軟性を確保することができる。
【００５３】
上記メモリ空間２００は、ハードウェア１００の上で動作するオペレーティングシステムで確保される仮想メモリ領域であり、各タスクに対応付けられた機密情報保持領域２０２～２０４と、オペレーティングシステムカーネルメモリ内に設けられ、前記機密情報保持領域２０２～２０４に対応付けられた鍵を保持する暗号化鍵保持用配列２０１を有する。
【００５４】
機密情報保持領域２０２～２０４は、実行中のタスクが、オペレーティングシステムカーネルによって休止状態に遷移させられる際に、当該タスクに対応する鍵によって暗号化され、休止状態のタスクが実行状態に遷移させられる際に、当該タスクに対応する鍵によって復号化される記憶領域である。
【００５５】
暗号化鍵保持用配列２０１は、当該オペレーティングシステムがサポートする最大タスク数（例えば、一般的なUNIX（登録商標）系オペレーティングシステムであれば６５５３６個）に等しい要素数を持つ暗号化鍵を保持するための記憶領域であり、この暗号化鍵保持用配列２０１は、オペレーティングシステムカーネルのみからアクセス可能とするようにアクセス権が設定されている。このアクセス権については、後述する。
【００５６】
そして、暗号化鍵保持用配列２０１に保持される個々の鍵は、対応するタスク番号を持つタスクのメモリ中でプログラミング時に指定される機密情報保持領域２０２～２０４を処理するために用いられる。具体的には、実行中のタスクがオペレーティングシステムカーネルによって休止状態に遷移させられる際に、当該タスクに対応付けられた機密情報保持領域（２０２，２０３，２０４等）が当該タスクに対応する鍵によって暗号化される。
【００５７】
逆に、休止状態から実行状態に遷移させられるタスクに関しては、当該タスクの休止処理時に暗号化された機密情報保持領域２０５を当該タスクに対応する鍵によって復号した上で、実行が再開されることになる。
【００５８】
図２は、本実施形態において、オペレーティングシステム起動時に必要とされる処理を示すフローチャート図である。ここでは具体例として４．４BSDオペレーティングシステムで行われる処理を挙げるが、本発明はこれに限定されるものではなく、他の方式のオペレーティングシステムでも利用することが可能である。
【００５９】
同図に示すように、オペレーティングシステム起動時にあっては、先ずブートプログラムと呼ばれる小規模なプログラムが起動され（Ｓ１０１）、そのブートプログラムによってオペレーティングシステム本体がメモリ上にロードされる。次いで、ＣＰＵの種類の同定や搭載メモリ容量の検査等のアセンブリ言語によって記述されたスタートアップルーチン

が実行され（Ｓ１０２）、Ｃ言語で記述されたオペレーティングシステム主要部分の実行準備が行われる。

【００６０】

その後、実行される機種に依存する初期化処理により、オペレーティングシステムが利用する各種システムデータの準備が行われる（Ｓ１０３）、これにより、暗号化鍵保持用の暗号化鍵保持用配列２０１が占めるメモリ領域は、この部分で割り当てられることになる。その後、機種に依存しない初期化処理を（Ｓ１０４）を行う。

【００６１】

ここで、オペレーティングシステムが実行されるマイクロプロセッサ１０１のメモリ管理ユニットが適切な機能（例えば、後述するＭＩＰＳ等）を持っている場合について説明する。

【００６２】

本実施形態において、暗号化鍵保持用配列２０１は、カーネル中の、機密情報保持に関連する部分が実行されている期間のみ当該領域を読み書きできる状態に設定する機能を備え、管理者権限で実行されるタスクであっても当該領域に対してアクセスできないように制限されている。

【００６３】

これにより、オペレーティングシステムがデバッグ等の目的で物理メモリに対するアクセスを可能とするようなインターフェースを備えている環境であっても、機密情報を保持することができる。例えば、４BSDではファイルシステム中に／dev／memという名前で物理メモリを読み書きするためのインターフェースが提供され、管理者権限をもったユーザタスクであれば、このインターフェースを利用して暗号化鍵保持用配列部分を読み出せるような場合であっても、本実施形態のように、暗号化鍵保持用配列２０１に対する読み書きについて制限を設けることで、機密情報が漏出のを回避することができる。

【００６４】

なお、上述した適切なメモリ管理ユニットを持つハードウェアの例としては、図３に示すような、マイクロプロセッサのひとつであるＭＩＰＳが挙げられる。

【００６５】

すなわち、同図に示すように、このＭＩＰＳアーキテクチャでは、マイクロプロセッサ１０１中に存在するハードウェアであるTLB（Translation Look－aside Buffer）１０１ａにより仮想アドレスと物理アドレスとの対応付けが行われる。この対応付けの際、具体的にどの仮想アドレスをどの物理アドレスにマッピングするかという決定はオペレーティングシステムカーネルによってなされる。

【００６６】

本実施形態では、このようなマッピングを管理するため、オペレーティングシステムカーネルは自らの管理するメモリ中にＴＬＢ管理領域２０６を確保し、維持する。このようなハードウェア上で本実施形態を用いる場合にはＴＬＢ管理領域２０６及びマイクロプロセッサ中のＴＬＢ１０１ａにおいて、秘密情報保持処理が実行されている間以外は暗号化鍵保持用の暗号化鍵保持用配列２０１に対してアクセスできないように設定しておく。これについて本実施形態におけるＭＩＰＳアーキテクチャの場合では、当該ページに対応するTLBエントリのValid（V）ビットをoffにすることで実現できる。

【００６７】

なお、本実施形態では、ＭＩＰＳマイクロプロセッサのアーキテクチャを具体例として挙げたが、本発明は、このような個別のマイクロプロセッサのみを対象とするものではなく、既存及び将来のマイクロプロセッサ全般に対して適用可能である。

【００６８】

一般に、仮想アドレス及び物理アドレス間のマッピングは、メモリ管理ユニットがサポートするページ単位で行われるため、このようなハードウェアによるアクセス管理を適用する場合、暗号化鍵保持用配列２０１は、ページ単位で確保する。

【００６９】

なお、本実施形態では、オペレーティングシステム起動時に行われるのは暗号化鍵保持用配列２０１に相当する領域の確保及びそのアクセス権設定操作のみであり、実際の暗号化鍵の生成は、タスク生成処理まで遅延される。

【００７０】

これにより重要な機密情報である当該暗号鍵の生成を真に必要となるまで遅らせることで最大限の安全性を確保することができ、また、タスク番号は再利用され得るデータであるため、個々の鍵の寿命をひとつのタスク限りとすることで長期間同じ鍵を使い続けることによる機密情報漏洩の可能性を低減させることが可能となる。なお、タスク番号の再利用を考慮する必要がない環境ではオペレーティングシステム起動時に必要な鍵を生成するようにしてもよい。

【００７１】

次いで、タスク生成処理時における暗号化鍵の生成処理について説明する。ここでは、システムコールに応じてタスクを生成する。新しくタスクを生成するための処理を図４に示す。

【００７２】

本実施形態では、４．４BSDを含むUNIX（登録商標）系オペレーティングシステムを前提とし、このUNIX（登録商標）系オペレーティングシステムにおけるタスクの生成は、forkシステムコールの実行によって実現される。forkシステムコールは新しく生成されるタスクのための管理用データをオペレーティングシステム内に割り当て、さらに同システムコールを実行したタスクの仮想メモリ空間の完全な複製を、新規タスク用のメモリ空間として確保することで新しいタスクの実行環境を構築する。

【００７３】

なお、本実施形態においてメモリ空間の複製の確保については、コピーオンライトとして知られるテクニックを用いることにより、古いタスクから複製された新しいタスクがメモリ領域に対してデータを書き込むまでは、実際に専用の物理メモリの確保は実行されない。一般に、forkシステムコールで生成された新しいタスクでは、続けてそれまで実行していたものとは異なる新しいプログラムコードをロードした上でゼロから実行を開始するため、このコピーオンライトのメカニズムを有効に実行することができる。

【００７４】

図４では、タスクｘが実行中にforkシステムコールを呼び出した結果として、新たにタスクｙが生成された状態を表している。同図に示すように、この状態においては、タスクｙ用の管理データ２１０が新しく生成され、当該管理データ２１０は、forkシステムコール呼び出し元であるタスクｘ管理データ２０９の機密情報保持領域２０８を、コピーオンライト型参照で共有している。

【００７５】

このようなタスク関連データ構造の他に、オペレーティングシステムカーネルメモリ中の暗号化鍵保持用配列２０１中には、新しいタスク番号ｙに相当する要素位置にタスクｙのメモリ中の機密情報保護のための鍵を新規に設定する。この鍵は、タスクｘのための鍵とは別途独立な異なる値であり、安全な擬似乱数データとしてforkシステムコール処理のなかで生成される。

【００７６】

この結果、機密情報保持領域２０８は、タスクｘとｙ双方によって共有されることになるが、当該領域を正しく復号できる鍵は同領域を生成したタスクｘの鍵のみであるため、forkシステムコールの結果生成されたタスクｙが同情報の内容を知ることはできない。このように親子関係にあるようなタスク間であっても、機密情報は当該情報を生成したタスクのみしかアクセスできないように規制される。

【００７７】

また、タスク管理データ２０９や２１０中にはタスクの実行コンテクストの一部として、コンテクスト切り替え時にハードウェア資源の一部である各種レジスタに保存されていた値を退避するレジスタ退避領域２０９ａや２１０ａが確保されている。

【００７８】
コンテクスト切り替え処理が発生した時点で機密情報の一部もしくは全部がこれらレジスタに格納されている可能性が存在するため、タスク管理データ中のレジスタ退避領域２０９ａや２１０ａに対しても、同様にタスク専用暗号化鍵で暗号化を行う。この場合、新しいタスクｙのレジスタ退避領域２１０ａに関してはタスクｘのレジスタ退避領域をコピーした後、当該領域をタスクｘの鍵で復号し、さらにタスクｙの鍵で暗号化する処理を行う。

【００７９】
さらに、本実施形態では、プログラマに対して機密情報保持領域を確保するために提供されるインターフェースが設けられている。図５はプログラマに対して機密情報保持領域を確保するために提供されるインターフェースの説明図である。

【００８０】
この機能は、基本的に、Ｃ言語において動的なメモリ領域割り当てを提供するライブラリ関数malloc()と同等のインターフェースを持つsallocシステムコールによって実現される。

【００８１】
すなわち、図５に示すように、このsallocシステムコールは、引数で与えられるサイズの連続したメモリ領域を新たにタスクのメモリ空間に割り付け（Ｓ２０１）、この割付けが成功したかを判断し（Ｓ２０２）、成功した場合には（図中ステップＳ２０２における”Ｙ”）、その部分が機密情報保持領域に関する情報（先頭アドレス及びサイズ）をカーネルメモリ空間中に記録する（Ｓ２０４）。その後、確保されたアドレス空間の次の領域のアドレスを返却する（Ｓ２０５）。ステップＳ２０２において、割付けが成功しなかった場合（図中ステップＳ２０２における”Ｎ”）、ＮＵＬＬ値を返却し（Ｓ２０３）、ステップＳ２０４以降の処理は実行しない。

【００８２】
そして、このように確保されたメモリ空間に対して暗号化及び復号化を実行する処理は、以下の手順により行う。図６及び図７は、コンテクストスイッチ処理発生時に行われる暗号化及び復号化処理の概要を示す説明図である。

【００８３】
先ず、暗号化を行う場合は、図６に示すように、コンテクストスイッチ処理により実行が中断させられ休止状態に遷移させられるタスクに関し、当該タスク用の機密情報暗号化鍵により退避されたレジスタ情報の暗号化を行った後（Ｓ３０１）、当該タスクの機密情報保持領域が存在するか否かについて判断を行い（Ｓ３０２）、機密保持領域が存在する場合（図中ステップＳ３０２における”Ｙ”）、その存在する機密保持領域がスワップファイルに退避されているか否かについて判断を行う（Ｓ３０４）。ステップＳ３０２において、機密保持領域が存在しないと判断した場合（図中ステップＳ３０２における”Ｎ”）は、そのまま処理を終了する（Ｓ３０３）。

【００８４】
ステップＳ３０４において、退避されていない領域が存在すると判断した場合（図中ステップＳ３０４における”Ｎ”）には、その領域を機密情報暗号化鍵によって暗号化する（Ｓ３０５）。一方、ステップＳ３０４において、当該領域がスワップファイルに退避されていると判断した場合（図中ステップＳ３０４における”Ｙ”）、上記ステップＳ３０２に戻り、上記同様の処理を繰り返す。

【００８５】
なお、ステップＳ３０４で、スワップファイルに退避されているかどうかを判定しているのは、最適化を図るためである。すなわち、このように判定することにより、機密情報保持領域すべて、一律に暗号化されるのを回避し、スワップファイルに退避されている情報を一時的に実メモリに読み込んだ上で暗号化処理を行うという重い処理を不要とすることができる。

【００８６】

なお、本実施形態において、当該データ部分をスワップファイルに退避する処理は、カーネルにしか行えないことから、当該タスクがまさに実行されているコンテクストの途中でスワップ退避処理が行われることはなく、当該タスクが休止状態になった後に、スワップ退避処理が実行される。

【００８７】

従って、本実施形態では、スワップ退避処理が実行される時点において、機密情報保持領域に対する暗号化による保護は実施されていることとなり、スワップファイルに退避される時点で機密情報保持領域は既に安全な状態となっており、そのようなスワップファイルに退避された機密情報保持領域に対しては、当該タスクの休止処理時における機密保持処理は、不要となる。

【００８８】

次いで、休止状態から実行状態へと遷移させられるタスクに関する復号処理について説明する。

【００８９】

図７に示すように、先ず、機密保持領域が存在するか否かについて判断を行い（Ｓ４０１）、機密保持領域が存在すると判断した場合（図中ステップＳ４０１における”Ｙ”）には、当該領域がスワップファイルに退避されているか否かについて判断を行う（Ｓ４０２）。当該ステップＳ４０２において、等が要ろう域がスワップファイルへ退避されている場合には、ステップＳ４０１に戻り、ステップ４０１以降の処理を繰り返す。

【００９０】

このステップＳ４０２において、スワップファイルに退避されていない領域が存在すると判断した場合（図中ステップＳ４０２における”Ｎ”）、当該タスクの機密情報保持領域のうち、先頭アドレスから、当該領域のサイズ分、機密情報暗号化鍵によって復号化を行う（Ｓ４０３）。

【００９１】

一方、ステップＳ４０１において、機密保持領域が存在しないと判断した場合（図中ステップＳ４０１における”Ｎ”）、当該タスク用の機密情報暗号化鍵により退避されたレジスタ情報の復号処理を行い（Ｓ４０４）、実行状態への遷移処理を終了する。

【００９２】

この復号処理においても、ステップＳ４０２において、機密情報保持領域中でスワップファイルへの退避が行われているかどうかの判定を行っているが、これも休止処理時の場合と同様に、最適化を図るためである。すなわち、上述した通り、スワップファイル上では機密情報保持領域は既に暗号化によって保護されているので安全性は問題ない。

【００９３】

また、既にスワップファイル上への退避処理が行われている場合には、当該領域に対するアクセスが比較的長期間行われていなかったこととなり、ここでタスクの実行が再開されたとしても当該領域に対するアクセスが行われるとは限らず、この時点でスワップファイルから実メモリに当該領域を読み込み、それを復号して当該タスクが利用できるように準備したとしても無駄となる可能性が高いのである。

【００９４】

実際に当該タスクがスワップファイルに退避させられている当該領域へのアクセスを行った場合には、ページフォールトが発生してオペレーティングシステムにより当該領域の実メモリへの読み込みが行われる。しかも、その処理自体は当然当該タスクを再度休止状態にした上で実施されることになるため、オペレーティングシステムによるスワップファイルから実メモリへのデータの読み込みが終了して改めて当該タスクが実行させられる時点になって再度コンテクストスイッチが発生し、その時点では当該機密情報保持領域はスワップファイルではなく実メモリに存在することとなり、ステップＳ４０４において、データの復号が行われることになる。

【００９５】

以上の結果、スワップファイル上に退避された機密情報保持領域の処置という例外的な事

象も、特別扱いすることもなくきわめて自然に処理できることになるのである。

【００９６】

［第２実施形態］

次いで、本発明の第５の実施形態について説明する。図８は、本実施形態に係るシステムの構成を示すブロック図である。本実施形態では、複数ＣＰＵのＳＭＰシステムに本発明を適用することを前提とする。すなわち、ＳＭＰシステムでは、複数のタスクが実際に同時並列的に実行することになるため、前述した第１実施形態における機能に加えて、機密情報保持領域に対して同時に動く他のタスクからのアクセスを禁止する機構を備えることを特徴とする。

【００９７】

なお、図８では、実際にタスク３とタスク４が同時に動いている状態を示している。この状態にあっては、機密情報保持領域２０５と２０６は復号されて平文状態であることから、本実施形態では、それぞれ他タスクからのアクセスが行えないようにアクセスを規制する。

【００９８】

具体的には、図９に示すように、このsallocシステムコールは、引数で与えられるサイズの連続したメモリ領域を新たにタスクのメモリ空間に割り付け（Ｓ５０１）、この割付けが成功したかを判断し（Ｓ５０２）、成功した場合には（図中ステップＳ５０２における”Ｙ”）、その部分が機密情報保持領域に関する情報（先頭アドレス及びサイズ）をカーネルメモリ空間中に記録する（Ｓ５０４）。ステップＳ５０２において、割付けが成功しなかった場合（図中ステップＳ５０２における”Ｎ”）、ＮＵＬＬ値を返却し（Ｓ５０３）、ステップＳ５０４以降の処理は実行しない。

【００９９】

ステップＳ５０４の後、実行環境がＳＭＰシステムであるかどうかを判定し（Ｓ５０５）、ＳＭＰである場合（図中ステップＳ５０６における”Ｙ”）には、割り当てた領域に対して、当該タスクが専有的に利用するような指定を行う（Ｓ５０６）。なお、このステップＳ５０６において、このような指定がハードウェア的なメモリ管理機構を用いて実現できる場合には、機密情報保持領域の割り当てを同メモリ管理機構が扱うページ単位とする。

【０１００】

一方、ステップＳ５０５において、実行環境がＳＭＰシステムでないと判断した場合（図中ステップＳ５０５における”Ｎ”）、或いは、ステップＳ５０６の後、確保されたアドレス空間の次の領域のアドレスを返却する（Ｓ５０７）。

【０１０１】

このような本実施形態に係るマルチタスク実行システムによれば、複数のＣＰＵを搭載するいわゆるＳＭＰ(Symmetric Multi Processor)構成の計算機において、同時に複数のタスクがそれぞれ別個のＣＰＵを占有することにより、真に同時並列的に実行が行うシステムであっても、あるタスクの実行中に、機密データ保持領域が平文のままという状態で存在するのを回避し、他のタスクからの操作により、当該領域が漏出するのを防止することができる。

【０１０２】

すなわち、本実施形態では、他のタスクによって機密データが漏出するのを防止するために、ＳＭＰ構成の計算機で用いる場合には、単にコンテクストスイッチ時に暗号化／復号化を行うだけではなく、実行中の機密情報を保持するメモリ領域が他のタスクからアクセスできないように制限することができる。

【０１０３】

具体的には、メモリ領域に対して特定のタスクのみからしか参照できないという属性の付与するとともに、機密情報保持用メモリ領域には同属性を付与する、といった機能をＳＭＰ対応オペレーティングシステムカーネルに追加する。

【０１０４】

なお、メモリデータにも一般的なファイルと同じく書き込み権限、読み出し権限、実行権限等の各種属性が存在し、そういった属性をサポートするために通常オペレーティングシステムにはあるメモリ領域がどのような属性を持っているのかを管理する機能が備えられていることから、既存のオペレーティングシステムの構成を大幅に変更することなく、上記属性を追加することは可能である。

【０１０５】

【発明の効果】

以上述べたように、この発明によれば、プログラマによって指定される機密情報保持領域を、当該プログラムが実行されているタスク以外のタスクからは読み出すことができないよう保護することが可能となり、機密情報を保持しながら実行するタスクをマルチタスク環境下で安全に実行させることができる。

【０１０６】

また、本発明によれば、保護機構は管理者権限を備えた悪意あるタスクに対してもアクセス制限を行うことができるため、管理者権限の漏洩や盗用という攻撃にも耐え得ることができる。

【図面の簡単な説明】

【図１】第１実施形態に係るマルチタスク実行システムの構成を示すブロック図である。

【図２】第１実施形態における起動時の処理を示すフローチャート図である。

【図３】第１実施形態において、適切なメモリ管理ユニットを持つハードウェア構成を備えた例を示す説明図である。

【図４】第１実施形態において、システムコールに応じて新しくタスクを生成するための処理を示す説明図である。

【図５】第１実施形態において、プログラマに対して機密情報保持領域を確保するために提供されるインターフェースの説明図である。

【図６】第１実施形態における、コンテクストスイッチ処理発生時に行われる暗号化処理の概要を示す説明図である。

【図７】第１実施形態における、コンテクストスイッチ処理発生時に行われる復号化処理の概要を示す説明図である。

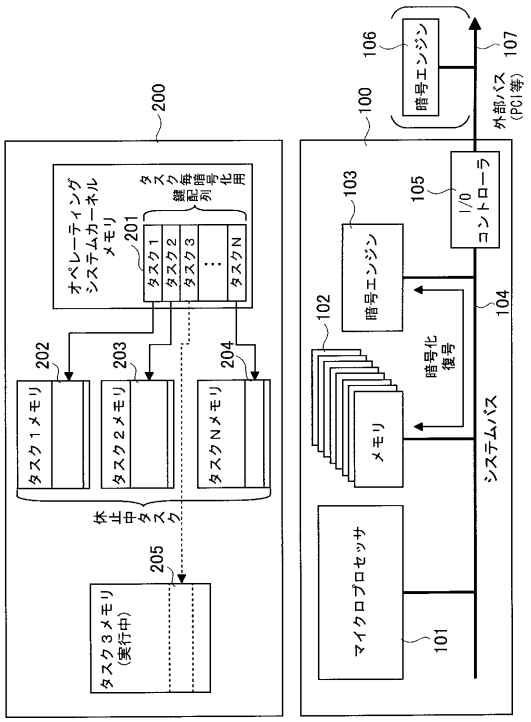
【図８】第２実施形態に係るシステムの構成を示すブロック図である。

【図９】第２実施形態における、マルチタスク実行システムの動作を示す説明図である。

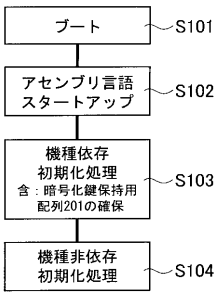
【符号の説明】

１００…ハードウェア
１０１…マイクロプロセッサ
１０１ａ…ＴＬＢ
１０２…メモリ
１０３…暗号処理エンジン
１０４…システムバス
１０５…コントローラ
１０６…増設ボード
１０７…外部バス
２００…メモリ空間
２０１…暗号化鍵保持用配列
２０２～２０５，２０８…機密情報保持領域
２０６…ＴＬＢ管理領域
２０９…タスク管理データ
２０９ａ…レジスタ退避領域
２１０…管理データ
２１０ａ…レジスタ退避領域

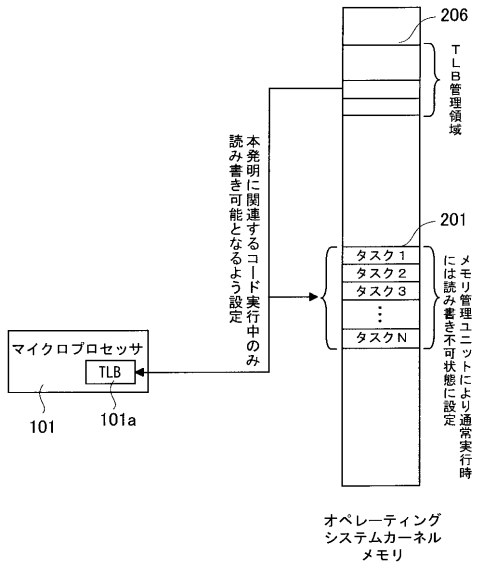
【図１】


200
100
106
暗号エンジン
107
外部バス
(PCI等)
オペレーティング
システムカーネル
メモリ
タスク毎暗号化用
鍵配列
201
タスク1
タスク2
タスク3
タスクN
103
暗号エンジン
105
I/O
コントローラ
102
暗号化
復号
104
メモリ
システムバス
202
タスク1メモリ
203
タスク2メモリ
204
タスクNメモリ
休止中タスク
205
タスク3メモリ
(実行中)
マイクロプロセッサ
101

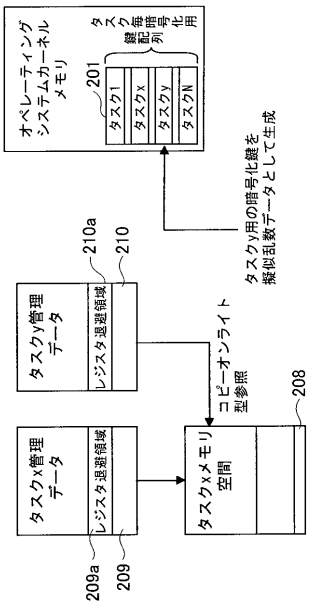
【図２】


ブート
S101
アセンブリ言語
スタートアップ
S102
機種依存
初期化処理
含：暗号化鍵保持用
配列201の確保
S103
機種非依存
初期化処理
S104

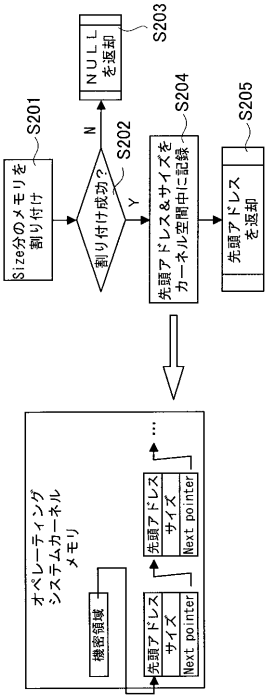
【図３】


206
TLB管理領域
201
タスク1
タスク2
タスク3
タスクN
メモリ管理ユニットにて通常実行時には読み書き不可状態に設定
本発明関連にて読み書き可能状態に設定
マイクロプロセッサ
TLB
101
101a
オペレーティング
システムカーネル
メモリ

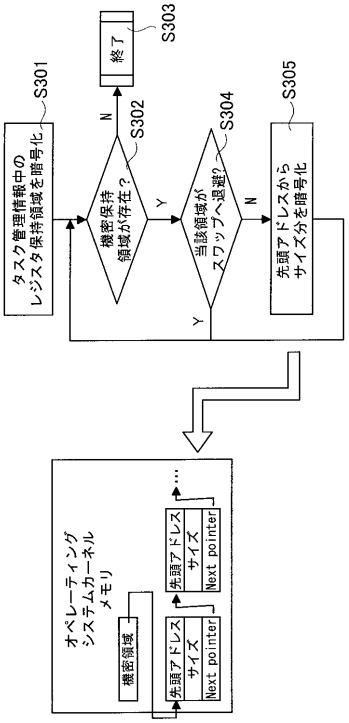
【図４】


オペレーティング
システムカーネル
メモリ
タスク毎暗号化用
鍵配列
201
タスク1
タスクx
タスクy
タスクN
タスクy用の暗号化鍵を
擬似乱数データとして生成
210a
210
タスクy管理
データ
レジスタ退避領域
コピーオンライト
型参照
208
タスクx管理
データ
レジスタ退避領域
タスクxメモリ
空間
209a
209

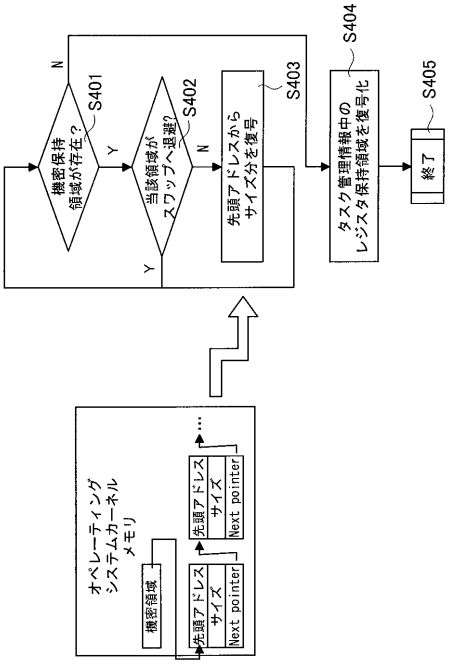
【図５】


Size分のメモリを割り付け
S201
割り付け成功？
S202
N
Y
NULLを返却
S203
先頭アドレス&サイズをカーネル空間中に記録
S204
先頭アドレスを返却
S205
オペレーティングシステムカーネルメモリ
機密領域
先頭アドレス
サイズ
Next pointer

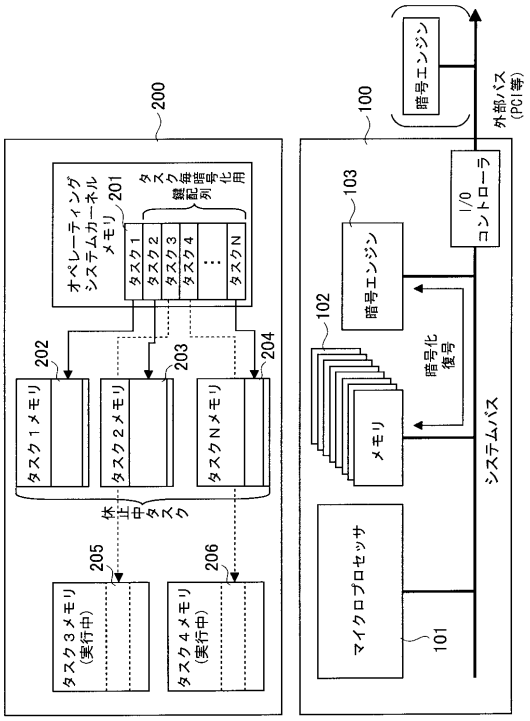
【図６】


タスク管理情報中のレジスタ保持領域を暗号化
S301
機密保持領域が存在？
S302
N
Y
終了
S303
当該領域がスワップへ退避？
S304
先頭アドレスからサイズ分を暗号化
S305
オペレーティングシステムカーネルメモリ
機密領域
先頭アドレス
サイズ
Next pointer

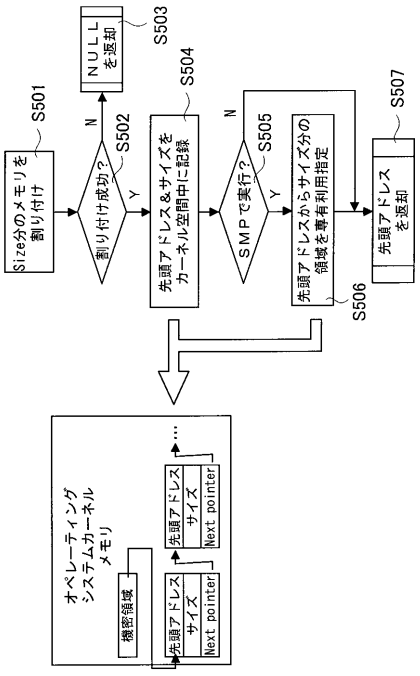
【図７】


機密保持領域が存在？
S401
N
Y
当該領域がスワップへ退避？
S402
N
Y
先頭アドレスからサイズ分を復号
S403
タスク管理情報中のレジスタ保持領域を復号化
S404
終了
S405
オペレーティングシステムカーネルメモリ
機密領域
先頭アドレス
サイズ
Next pointer

【図８】


200
オペレーティングシステムカーネルメモリ
201
タスク1
タスク2
タスク3
タスク4
タスクN
タスク毎鍵暗号化用
202
203
204
タスク1メモリ
タスク2メモリ
タスクNメモリ
休止中タスク
205
206
タスク3メモリ（実行中）
タスク4メモリ（実行中）
100
暗号エンジン
外部バス（PCI等）
I/Oコントローラ
103
102
暗号エンジン
メモリ
暗号化復号
システムバス
マイクロプロセッサ
101

【図９】


Size分のメモリを割り付け
S501
割り付け成功？
S502
N
Y
NULLを返却
S503
先頭アドレス＆サイズをカーネル空間中に記録
S504
SMPで実行？
S505
N
Y
先頭アドレスからサイズ分の領域を専有利用指定
S506
先頭アドレスを返却
S507
オペレーティングシステムカーネルメモリ
機密領域
先頭アドレス
サイズ
Next pointer
先頭アドレス
サイズ
Next pointer

フロントページの続き

(56)参考文献 特開２００１－３１８７８７（ＪＰ，Ａ）
特開２００３－０５１８１９（ＪＰ，Ａ）
特開２００２－１４０２３６（ＪＰ，Ａ）
特開２００２－２３２４１７（ＪＰ，Ａ）
特開平０９－２５８９７７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 12/14
G06F 9/48
G06F 21/24