(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4907603号
(P4907603)
(45)発行日　平成24年4月4日(2012.4.4)　(24)登録日　平成24年1月20日(2012.1.20)

(51)Int.Cl.　　F I
G06F 13/00 (2006.01)　　G06F 13/00　353C
G06F 21/24 (2006.01)　　G06F 21/24　163A

請求項の数 13　外国語出願　(全 20 頁)

(21)出願番号　特願2008-161206(P2008-161206)
(22)出願日　平成20年6月20日(2008.6.20)
(65)公開番号　特開2009-9566(P2009-9566A)
(43)公開日　平成21年1月15日(2009.1.15)
審査請求日　平成20年8月18日(2008.8.18)
(31)優先権主張番号　1380/CHE/2007
(32)優先日　平成19年6月27日(2007.6.27)
(33)優先権主張国　インド(IN)

前置審査

(73)特許権者　511076424
ヒューレット－パッカード　デベロップメント　カンパニー　エル．ピー．
Ｈｅｗｌｅｔｔ-Ｐａｃｋａｒｄ　Ｄｅｖｅｌｏｐｍｅｎｔ　Ｃｏｍｐａｎｙ，　Ｌ．Ｐ．
アメリカ合衆国　テキサス州　７７０７０　ヒューストン　コンパック　センタ　ドライブ　ウエスト　１１４４５
(74)代理人　110000039
特許業務法人アイ・ピー・エス

最終頁に続く

(54)【発明の名称】アクセス制御システムおよびアクセス制御方法

(57)【特許請求の範囲】
【請求項１】
１つ以上のコンパートメントを定義し、このコンパートメントに関連付けられているエンティティによるネットワークサービスへのアクセスを制御するために、前記１つ以上のコンパートメントに適用される規則を提供するアクセス制御システムであって、前記規則は、少なくとも、動的に割り当てられる通信ポートを使用するネットワークサービスへのアクセスを制御するための第１の種類の規則を含み、
当該システムは、
コンピュータによって実行され、エンティティからのネットワークサービスコールを受信するネットワークサービスハンドラと、
前記コンピュータに格納され、前記コンパートメントとネットワークサービスとを対応付けるテーブルと、
前記コンピュータによって実行され、ネットワークサービスの前記規則を提供する、各ネットワークサービスに対応する１つ以上のアクセスチェックモジュールと、
前記コンピュータによって実行され、前記ネットワークサービスコールに基づいて前記テーブルを検索し、前記コンパートメントに関するネットワークサービスに対応する前記アクセスチェックモジュールの１つを識別するセキュリティコンテインメントコントローラと
を備え、
前記規則は、１つのコンパートメントについて、第１のネットワークサービスへのアク

セスを許可し、前記第１のネットワークサービスとは異なる第２のネットワークサービスへのアクセスを拒否するように定義され、
前記識別されたアクセスチェックモジュールは、前記規則に基づいて、ネットワークサービスへのアクセスを許可または拒否する
システム。
【請求項２】
前記規則は、事前に割り当てられている通信ポートを使用するネットワークサービスへのアクセスを制御するための第２の種類の規則を含む
請求項１に記載のシステム。
【請求項３】
前記第１の種類の規則と、前記第２の種類の規則とは異なる
請求項２に記載のシステム。
【請求項４】
前記第１の種類の規則は、いずれの通信ポートとも無関係に定義される
請求項１に記載のシステム。
【請求項５】
前記第１の種類の規則は、ネットワークサービス識別子に関して定義される
請求項１に記載のシステム。
【請求項６】
前記適用される規則は、前記システムの規則テーブルにおいて識別される
請求項１に記載のシステム。
【請求項７】
前記適用される規則は、モジュールにおいて定義され、
前記規則テーブルは、適用される各規則を識別し、それぞれの前記モジュールへの関連するポインタを含む
請求項６に記載のシステム。
【請求項８】
前記ネットワークサービスハンドラは、ネットワークサービスコールを受信し、適用されるアクセス制御規則に関して、前記コール側エンティティが、前記識別されたネットワークサービスにアクセスするための許可を有するか否かを確認するように、このシステムに指示し直し（redirect）、適用されるアクセス制御規則が前記ネットワークサービスにアクセスする許可を付与する場合にのみ、前記ネットワークサービスコールを処理するように構成されている
請求項１に記載のシステム。
【請求項９】
前記ネットワークサービスハンドラは、ネットワークサービスコールを処理するrpc_call()機能
を備える
請求項１に記載のシステム。
【請求項１０】
前記コール側エンティティが、前記ネットワークサービスにアクセスする権利を有するか否かを確認するための関数をコールする関数コール
を含む請求項９に記載のシステム。
【請求項１１】
強制アクセス制御の規則を適用して、ネットワークサービスへのアクセスを制御する
請求項１に記載のシステム。
【請求項１２】
セキュアオペレーティングシステム
を含む請求項１に記載のシステム。
【請求項１３】

ＲＰＣネットワークサービスへのアクセスを制御するアクセス制御規則を提供する
請求項１に記載のシステム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、コンピュータシステムにおけるアクセス権に関する。
【背景技術】
【０００２】
オペレーティングシステムは、通常、コンピュータのその後のオペレーションを制御するためにコンピュータのメモリにロードされて実行されるコンピュータプログラムまたはコンピュータプログラムのセットである。
しかしながら、オペレーティングシステムは、特に、たとえば移動電話およびＰＤＡ等のポータブルデバイスでは、ファームウェアまたはハードウェアの組み込みプログラムとすることもできる。
【０００３】
ほとんどの従来のコンピュータオペレーティングシステムは、アクセス制御の形で、或る種の論理的保護をデータに提供する。
この論理的保護は、特定の人々または特定の人々のグループに対して付与または拒否することができる。
一般に、任意アクセス制御（ＤＡＣ）を提供するシステムでは、ユーザ（アドミニストレータと相対する者として）は、自身のデータに許可を割り当てることができる。
これによって、他者（または他者のグループ）のそのデータへのアクセスが許可または拒否される。
これは、個人にとっては満足のいくものである。
しかしながら、いくつかの組織、特に軍隊組織または政府機関等では、情報へのアクセスをより厳重に制御する能力が必要とされる。
たとえば、最高機密情報は、組織内のほとんどの人に見えるものであるべきではなく、制限情報は、そのラベルが示唆するように、一般に利用可能であるべきではないのに対し、非制限情報は、組織内のだれによるアクセスにも利用可能とすることができる。
【０００４】
したがって、組織の情報に対してより優れたアクセス制御を提供するセキュアオペレーティングシステムが知られている。
通常、セキュアオペレーティングシステムは、付加的な分類またはラベルをファイルに関連付け、いわゆる強制アクセス制御（ＭＡＣ）を適用する。
ＭＡＣは、（たとえば、機密ラベルによって表されるような）ファイルの機密性に基づいてファイルへのアクセスを制限する手段を提供する。
ＤＡＣとは対照的に、ＭＡＣの下では、ユーザは、誰が自身のデータを見るのかを決定する権利を有しない。
すなわち、互換性のある情報取扱許可（clearance）を有するユーザのみが、データを見ることを許可される。
たとえば、最高機密情報取扱許可を有するユーザは、それよりも下位の情報取扱許可を有する他者が自身のデータを見ることを許可する能力を有しない。
【０００５】
ＭＡＣは、「コンパートメント」によって表すことができる。
実際には、コンパートメントは、通常、名称等の識別子を有する論理構成体（logical construct）であって、そのコンパートメントを定義する一組のアドミニストレータが構成したアクセス制御規則が適用される論理構成体である。
コンパートメント規則は、アプリケーションの実行に必要な資源（ファイル、プロセス、プロセス間通信メソッド等）のみへのアクセスを許可するのに使用される。
これらの規則は、コンパートメントにおいて動作する許可を有するユーザおよびプロセ

スの双方に適用され、したがって、別段の指定のない限り、または、文脈上別の意味が指示されていない限り、このようなユーザおよびプロセスは、本明細書では「エンティティ」と総称される。

【０００６】

したがって、コンパートメント内で動作するエンティティは、デフォルトとは逆の特定のＭＡＣ規則が提供されない限り、デフォルトでは、同じコンパートメントにおいてアクセス可能であると定義されたファイル、他のプロセスおよび資源にしかアクセスすることができない。

【０００７】

コンパートメントは、アプリケーションについて隔離されたランタイム環境を提供することができる。

この隔離されたランタイム環境では、コンパートメント規則によって、アプリケーションの実行に必要な資源のみへのアクセスが可能になる。

これによって、たとえアプリケーションが損なわれていても、その損傷が、そのアプリケーションが実行されているコンパートメント（複数可）にのみ限定される可能性が増加する。

【０００８】

ファイルシステム等へのアクセスを制御することに加えて、ＭＡＣ規則を定義することによって、ＭＡＣが、一般のリモートサービス、より詳細にはネットワークサービスを提供するネットワークエンドポイント等、コンパートメントの外部の資源へのアクセスを制御するのに使用されることが知られている。

これらのＭＡＣ規則は、コンピューティングプラットフォームの特定の通信インターフェースへのアクセスを許可または拒否する。

本明細書で使用される場合、「ネットワークサービス（またはネットワーク接続サービス）」という用語は、ネットワーク接続された複数のコンピュータから成る分散コンピューティング環境においてリモートコンピュータ、すなわちサーバ上で実行されるコンピュータプログラムである。

ネットワークサービスは、サービスを実行するホストから遠隔にあるクライアント（たとえば、セキュアオペレーティングシステムを実行しているホスト）からアクセスして起動する、すなわち「呼び出す」ことができる。

ネットワークサービスの例は、既知のサービスを２～３例を挙げると、ＲＰＣサービス（ypserv、mountd等）、ウェブサイト、Ｘ端末（X-terminal）、およびＦＴＰサイトである。

【０００９】

ネットワークサービスへのアクセスを制御する能力は、既に損なわれているおそれのあるネットワークサービスまたは今後損なわれるおそれのあるネットワークサービスにエンティティがアクセスすることを防止するためのメカニズムを提供することによって、セキュリティを増加させる。

ネットワークサービスが損なわれる可能性のある最も一般的な様態は、サービス拒否攻撃によるものである。

この攻撃は、適法なユーザに対してサービスを利用不能にしようとして、サービスのダミー要求でサービス（またはサービスを実行しているサーバ）をフラッドさせるものである。

【００１０】

資源アクセスのためのＭＡＣを定義する１つの既知の方法は、いわゆるラベル化セキュリティ保護プロファイル（labelled security protection profiles）（ＬＳＰＰ）に基づくものである。

このＬＳＰＰでは、資源（たとえば、ネットワークエンドポイント）は、資源上のアクセス制御ポリシーを指定するラベルを保持するように構成されている。

しかしながら、ＭＡＣをこのように実施するには、多大な労力と、従来のアプリケーシ

ョンおよびシステムへの適応が必要とされる。
資源アクセスのためのＭＡＣを定義する既知のより簡単な方法は、アクセス制御規則を使用することに基づくものである。
アクセス制御規則は、個々のコンパートメントごとに指定される。
このような制御規則は、通常、ＴＣＰポート番号を使用して、ネットワークエンドポイント、たとえばネットワークサービスを一意に識別する。
多くのネットワークサービスは、事前に割り当てられている既知のポート番号を使用する。
たとえば、ＨＴＴＰはＴＣＰポート８０を使用し、ＰＯＰ３はポート１１０を使用し、ＩＲＣはポート１９４を使用する。
このような場合に、ＭＡＣ規則を使用してそれぞれのポートへのアクセスを付与または拒否することによって、サービスへのアクセスを付与または拒否することが可能である。
しかしながら、ＲＰＣサービスおよびＸＴＥＲＭサービス等のいくつかのネットワークサービスは、事前に割り当てられているポート番号を使用せず（すなわち、ポート番号は、ランタイムにおいて動的に配分、すなわち割り当てられ）、したがって、コンパートメント規則でポート番号を指定することによってアクセスを付与または拒否することが可能でない。
このようなアクセス制御がないことから、いずれかのこのようなネットワークサービスが損なわれた場合に、通常はセキュアであるシステムに与えられる可能性のある損傷を抑制または制御することが容易でない場合がある。

【発明の開示】
【発明が解決しようとする課題】
【００１１】
従来技術の問題の１つ以上を少なくとも軽減することが、本発明の実施の形態の目的である。
【課題を解決するための手段】
【００１２】
本発明の一態様によれば、１つ以上のコンパートメントを定義すると共に、当該コンパートメントに関連付けられるエンティティによるネットワークサービスへのアクセスを制御するための、コンパートメント（複数可）に適用される規則を提供するアクセス制御システムであって、規則は、動的に割り当てられる通信ポートを使用するネットワークサービスへのアクセスを制御するための少なくとも第１の種類の規則を含む、システムが提供される。
【００１３】
本発明の別の態様によれば、アクセス制御方法であって、
セキュアオペレーティングシステムのコンパートメントに関連付けられる要求側エンティティからのネットワークサービスコールを受信することであって、上記ネットワークサービスは、動的に割り当てられる通信ポートを使用する、受信すること、
コンパートメントに適用されているあらゆる規則に関して、エンティティがネットワークサービスにアクセスするための許可を有するか否かを判断すること、および
適用されるコンパートメント規則が、ネットワークサービスにアクセスするための許可を付与する場合にのみ、ネットワークサービスコールを処理することを含む、方法が提供される。
【００１４】
本発明のさらに別の態様によれば、ネットワークサービスコールを処理するように構成されているデータ処理システムであって、
該システムのコンパートメントに関連付けられるエンティティから、動的に割り当てられる通信ポートを使用するネットワークサービスのコールを受信するための入力と、
コンパートメントに適用されると共に関連するネットワークサービスアクセス許可を定義する規則を記憶するストアと、

ストアが、ネットワークサービスにアクセスするための許可を付与するコンパートメントの規則を含む場合にのみ、ネットワークサービスコールを実行に移すプロセスとを備える、システムが提供される。

【００１５】

既知のセキュアオペレーティングシステムは、ＳＥＬｉｎｕｘ（商標）、Ｔｒｕｓｔｅｄ　Ｓｏｌａｒｉｓ（商標）、およびＨＰ－ＵＸ　Ｓｅｃｕｒｉｔｙ　Ｃｏｎｔａｉｎｍｅｎｔ（商標）であり、本発明の態様の実施の形態は、これらのオペレーティングシステムに適用することができる。ただし、教示される原理は、より広く適用可能である。

【００１６】

本発明のさらなる特徴および利点は、単なる例として与えられた、本発明の好ましい実施の形態の以下の説明から明らかになる。以下の説明は、添付図面を参照して為される。

【発明を実施するための最良の形態】

【００１７】

既知のポート番号を使用するネットワークサービスへのアクセス（またはその使用）を制御することが可能であることが知られているが、本発明の実施形態は、新しいクラスのコンパートメント規則に関するものであり、この新しいクラスのコンパートメント規則は、動的に割り当てられるポート番号を使用するネットワークサービスへのアクセス（またはその使用）を制御するものである。

【００１８】

次に、本発明の実施形態を詳細に説明する前に、本発明の特定の実施形態を実施するのに適したコンピューティング環境を説明する。

【００１９】

本発明の実施形態を実施するのに適した例示的なコンピューティングプラットフォーム１０を図１の図解に示す。

このコンピューティングプラットフォーム１０は、キーボード１４、マウス１６、および視覚表示ユニット（ＶＤＵ）１８の標準的な機能を有する。

これらは、プラットフォームの物理的な「ユーザインターフェース」を提供する。

プラットフォームは、マザーボード、および、ファイルシステムを記憶するためのハードディスクドライブ等の標準的なコンポーネントを含む。

【００２０】

図２に示すように、コンピューティングプラットフォーム１０のマザーボード２０は、（他の標準的なコンポーネントの中でも特に）メインプロセッサ２１、メインメモリ２２、データバス２６およびそれぞれのアドレス指定ライン２７および制御ライン２８、プラットフォーム１０のＢＩＯＳプログラムを含むＢＩＯＳメモリ２９、並びに入出力（ＩＯ）コントローラ２５を含む。

ＩＯコントローラ２５（１つ以上の独立したサブシステムを備えることができる）は、マザーボードのコンポーネントと接続されているデバイスとの間の相互作用を制御する。

接続デバイスは、（入力コントローラ２５３を介した）キーボード１４およびマウス１６、（出力コントローラ２５４、たとえば、スクリーンコントローラまたはグラフィックスコントローラを介した）ＶＤＵ１８、（ストレージコントローラ２５２を介した）ハードディスク等のストレージデバイス１２、並びに（ネットワークコントローラ２５１、たとえば、イーサネット（登録商標）ＬＡＮコントローラを介した）外部世界等である。

ネットワークコントローラ２５１は、ネットワークインターフェースカード（ＮＩＣ）上に具現化することができる。

たとえばインターネット全体にわたる外部世界との通信は、ネットワークコントローラ２５１を介して方向付けられ、通常、データパケットを含む。

既知のシステムでは、以下でより詳細に説明するように、データパケットは、通常、ＮＩＣ等の論理ポート番号を介して移動するようにアドレス指定される。

【００２１】

メインメモリ２２は、通常、ランダムアクセスメモリ（ＲＡＭ）であり、ＢＩＯＳメモ

リは、通常、ＥＥＰＲＯＭである。
ＥＥＰＲＯＭは、ＢＩＯＳ更新で「フラッシュ」することができる。
動作中、プラットフォーム１０は、ＢＩＯＳメモリ２９からＲＡＭ内へＢＩＯＳをロードし、次に、ハードディスク１２からＲＡＭ内へオペレーティングシステムをロードする。
オペレーティングシステムは、コンピューティングプラットフォームのオペレーションのあらゆる態様を制御し、本発明例では、コンパートメントを提供するセキュアオペレーティングシステムである。
加えて、動作中、オペレーティングシステムは、ハードディスク１２からＲＡＭ２２内へ、プラットフォーム１０によって実行することができるプロセスまたはアプリケーションもロードする。

【００２２】

図３の図解は、マシン１（３００）、マシン２（３１０）、およびマシン３（３２０）のラベルが付けられた３つのネットワーク接続されているコンピュータが存在するネットワークシナリオを示している。
マシン１は、ＲＰＣネットワークサービスrpcbind３０１、ypserv３０２、およびrpc.mountd３０３を実行するサーバである。
これらのＲＰＣサービスは、ネットワークカード３０５に対して動的に割り当てられるポート番号３０４を使用する。
マシン２は、セキュアオペレーティングシステム３１２を実行している。
このセキュアオペレーティングシステム３１２は、指定された「ファイル共有」３１３、「アイフェース」（ifaces）３１４、および「データベース」３１５の３つのコンパートメントを定義する。
これらのコンパートメントは、ネットワークカード３１４のポート３１５を使用して通信することができる。
マシン３は、ウェブサーバ３２２を実行している。
ウェブサーバ３２２は、ネットワークカード３２４に対して事前に割り当てられているポート番号８０（３２５）を使用する。
図解の斜線領域は、たとえばＬＡＮによって提供される、マシン間の通信ファブリックまたは通信インフラストラクチャを表している。
第１の斜線領域３３０は、マシン１およびマシン２が原則として互いに通信できることを示し、第２の斜線領域３４０は、マシン２およびマシン３が原則として互いに通信できることを示している。
しかしながら、通信を行うには、（パケットベースまたはストリームベースのいずれかの）論理接続が確立されなければならない。
マシン２のポート８１とマシン３のポート８０との間で通信を行うことを可能にするこのような１つの論理接続は、３５０として指定された実線矢印によって示されている。

【００２３】

既知のセキュアオペレーティングシステムを使用して、マシン２のコンパートメントで動作するエンティティによる、マシン３のウェブサーバ３２２へのアクセスを制御することが可能である。
この既知のセキュアオペレーティングシステムは、固定されたポート番号割り当てと、ポート番号（すなわち、より具体的には、ＮＩＣ、宛先ＩＰアドレス、およびポート番号）に基づくコンパートメント特有の規則とを使用するネットワークサービスへのアクセスを制御することができる。
次に、背景として、この種の例示的な既知の規則を説明する。
この規則は、次のシンタックスを有する。
「アクセス，方向，プロトコル，ポート（ポート番号），ピアポート（ポート番号），コンパートメント」
ここで、アクセスは、付与または拒否に設定される。

方向は、規則がクライアント（コンパートメントからのアウトバウンドメッセージ）に適用されるのか、またはサーバ（コンパートメントへのインバウンドメッセージ）に適用されるのかを指示する。

プロトコルは、たとえば、ＴＣＰまたはＵＤＰ（または他の任意の認識されているプロトコル）として設定することができる。

ポートは、コンパートメントを含むマシンのポートである。

ピアポートは、サービスを含むマシン（またはネットワークエンドポイント）によって使用されるポートである。

コンパートメントは、通信に使用されるネットワーク（またはＬＡＮ）インターフェースに関連付けられているコンパートメントの識別子または名称である。

たとえば、「ファイル共有」と呼ばれる、コンパートメントのアクセス規則は以下のように表され得る。

［従来技術のＭＡＣ規則］

```
10    compartment File_Sharing {
20        grant server tcp port 81 peer port 80 ifaces
30        deny client tcp port 81 peer port 80 ifaces
40    }
50    compartment ifaces {
60        interface lan0
70    }
```

【００２４】

このアクセス規則は、以下の意味を有する。

すなわち、ライン１０は、コンパートメントをFile Sharing（ファイル共有）と命名している。

ライン２０は、（サーバによって指定される）リモートコンピュータのＴＣＰポート８０からＴＣＰポート８１へのＴＣＰプロトコルインバウンド通信を受信するためのアクセス制御権をコンパートメントFile Sharingに付与している。

ＴＣＰポート８１は、インターフェースカードが指定するlan0（図示せず）に関連付けられている。

lan0は、それ自体ifaces（アイフェース）と呼ばれるコンパートメントに関連付けられている。

ライン３０は、ＴＣＰポート８１からlan0インターフェースカードを通じてリモートコンピュータのＴＣＰポート８０へＴＣＰ通信を送出するためのアクセス制御権を（クライアントによって指定される）コンパートメントFile Sharingに付与する。

ライン４０は、File Sharingコンパートメントの定義を終了する。

ライン５０は、ifacesと呼ばれるコンパートメントの定義を開始する。

ライン６０は、（File Sharingコンパートメントにライン２０および３０で付与されたように、他の任意のコンパートメントのエンティティが、ifacesコンパートメントにアクセスする許可を有する場合に、lan0インターフェースを介してのみ通信できるように）lan0インターフェースを含むようにifacesコンパートメントを定義する。

ライン７０は、ifacesコンパートメントの定義を終了する。

ポート８０は、ＨＴＴＰに関連付けられるＴＣＰポートであることが知られている。

ＨＴＴＰは、ウェブページを転送するのに使用され、したがって、File Sharingコンパートメントは、上述のアクセス制御規則によって、ウェブサーバと相互作用する許可を付与されている。

【００２５】

マシン１のポート３０５が動的に割り当てられ（したがって、それぞれのポート番号は、未知であるとして「？」のラベルが付けられ）、規則を適用することができないとき、ファイル共有コンパートメント３１３とウェブサーバ３２２との間のアクセスを制御することができるのとは対照的に、マシン１によってホスティングされているＲＰＣサービス

へのファイル共有コンパートメント３１３によるアクセスを付与することは（上述したクラスの規則を使用して）可能でなく、同時に、マシン１のＲＰＣサービスへのデータベースコンパートメント３１５によるアクセスを拒否することも可能でない。

【００２６】

一方、本発明の特定の実施形態によれば、説明するように、コンパートメントのそれぞれについて独立したネットワークサービスアクセス制御を達成することができる。

本明細書で詳細に説明するように、本発明の実施形態は、動的に割り当てられるポート割り当てを使用するアクセスネットワークサービスを制御するように設計されている新しいクラスの規則に依存する。

この新しい規則は、例として、以下のように表すことができる。

DPR G/D <MODULE> <STRING>

ここで、DPRは、「Dynamic Port Rule」（動的ポート規則）を表す規則名である。

G/Dは、許可ＧＲＡＮＴ（付与）またはＤＥＮＹ（拒否）である。

<MODULE>は、規則が適用されるネットワークサービス（および対応するモジュールまたは関数、説明する）の名称である。

<STRING>は、命名されたネットワークサービス（および対応するモジュールまたは関数）に関連付けられている一意の識別子（番号、名称、または他の任意の種類の参照子とすることができる）である。

【００２７】

図４の図解は、例示的なセキュアオペレーティングシステムの一部の高レベルの機能表現を提供する。

このセキュアオペレーティングシステムは、本発明の実施形態によるオペレーションに適合されている。

本発明の実施形態は、図４に示す配置にもアーキテクチャにも決して限定されるものではないことを強調しておく。

本発明の特定の実施形態を実施することができるセキュアオペレーティングシステムの一例は、ＨＰ－ＵＸ　Ｓｅｃｕｒｉｔｙ　Ｃｏｎｔａｉｎｍｅｎｔのバージョン２、リリース１１．３１である。

このＨＰ－ＵＸ　Ｓｅｃｕｒｉｔｙ　Ｃｏｎｔａｉｎｍｅｎｔは、ＭＡＣを提供するものであり、本明細書で説明するように、ネットワークサービスアクセス規則を、コンパートメント単位で、動的に割り当てられるポートを使用するネットワークサービスに提供するように適合することができる。

しかしながら、本発明の特定の実施形態は、他の既存のセキュアオペレーティングシステム上で実施することができ、本明細書で説明する原理は、ＨＰ－ＵＸ製品のアプリケーションに決して限定されるものではないことが認識されよう。

【００２８】

以下の例では、オペレーティングシステムは、説明するように、コンパートメント規則を使用して（ネットワークサービスに関連して）実施される、ＭＡＣを提供するコンパートメントベースのオペレーティングシステムである。

オペレーティングシステムは、ＵＮＩＸ（登録商標）のようなコマンドと、既知のポート番号を有するネットワークサービスへのアクセスを制御するための既存の規則と、動的に割り当てられるポート番号を有するネットワークサービスへのアクセスを制御するための新しい規則とをサポートする。

この場合も、本発明の実施形態は、ＵＮＩＸ（登録商標）のようなオペレーティングシステムに決して限定されるものではない。

たとえば、これらの原理は、Ｍｉｃｒｏｓｏｆｔ（商標）Ｗｉｎｄｏｗｓ（登録商標）ベースのオペレーティングシステムにも等しく適用することができるし、さらには、ｂｅｓｐｏｋｅまたは組み込みオペレーティングシステム等を含む、他の任意の種類のオペレーティングシステムにも適用することができる。

【００２９】

図４のオペレーティングシステム４００は、カーネル空間４０２およびユーザ空間４０４を有するものとして示されている。
カーネル空間４０２は、オペレーティングシステムのカーネルを実行するために確保されたメモリである。
カーネルは、一般に、コンピューティングプラットフォームの資源の管理、および、ハードウェアコンポーネントとソフトウェアコンポーネントとの間の通信の管理を担当する。
ユーザ空間４０４は、ユーザアプリケーションが実行されて動作するメモリエリアである。
ユーザ空間４０４は、（時に「標準入力」と呼ばれるマウス１６またはキーボード１４を介して）ユーザからの入力を受け取り、その入力をシステムコールインターフェース４０６に引き渡す。
本発明の文脈では、この入力が、或る種のネットワークサービス要求に関するものであるとき、コマンドが、システムコールインターフェース４０６によってネットワークサービスハンドラ４０８へ渡される。
【００３０】
ネットワークサービスハンドラ４０８は、オペレーティングシステムにおける（または、オペレーティングシステムによってコールされる）ソフトウェアプログラムまたは関数である。
このソフトウェアプログラムまたは関数は、セキュリティコンテインメントコントローラ（Security Containment controller）４１０によって提供されるアクセス制御規則を参照してネットワークサービスコールを処理する。
セキュリティコンテインメントコントローラ４１０は、セキュリティコンテインメント規則関数４１２内にＭＡＣ規則を含む。
セキュリティコンテインメント規則関数４１２は、システムのコンパートメントのそれぞれを定義する一般の規則を含む。
特に、セキュリティコンテインメント規則関数４１２は、本発明の本実施形態によれば、以下でより詳細に説明するように、コンパートメントのネットワークサービスアクセス制御を定義する規則を含む。
【００３１】
ネットワークサービスハンドラ４０８は、コンパートメントに関連付けられている、ネットワークサービスコールを発行したエンティティ（すなわち、ユーザまたはプロセス）がコールの起動および完了に必要なアクセス制御権を有するか否かを、セキュリティコンテインメントコントローラ４１０を参照することによって確認する。
エンティティが権利を有する場合には、ネットワークサービスハンドラ４０８は、既知の方法で関連するネットワークサービスコールを実行し、ＬＡＮカードコントローラ４１４（ＢＩＯＳコールを使用する）と相互作用して、ＬＡＮカード４１６およびＬＡＮ３３０を介してそれぞれのネットワークサービスサーバ３００と相互作用する。
エンティティが権利を有しない場合には、アクセスは従来の方法で拒否される。
【００３２】
特にネットワークサービスアクセス制御規則に関しては、図４に示すように、セキュリティコンテインメント規則関数４１２は、アクセス制御許可を割り当てられる各ネットワークサービスについて、識別されるコンパートメントごとに、エントリー４２０を含むテーブル４１８を含む。
示される例では、「ＦＳ」コンパートメントの１つのテーブルエントリー４２０しか示されていないが、実際には、アクセス制御が必要とされるさまざまなコンパートメントについて多くの規則が存在し得ることが認識されよう。
原則として、コンパートメントは特定の規則を一切必要とせず、その場合に、デフォルトの姿勢は、コンパートメントで動作するエンティティがそれぞれのネットワークサービスへのアクセスを拒否されるということである。

【００３３】
図４に示すように、コンパートメント規則テーブル４１８は、コンパートメント名またはコンパートメントＩＤの列４２０と、モジュール名の列４２２と、ポインタの列４２４とを含む。
ここで、テーブルの各行（すなわち、識別されるコンパートメントの各モジュール名のエントリー）は、対応するポインタを有する。
この例では、（ポインタの列４２４の）ポインタ４２６は、好ましくは、「不透明ポインタ」である。
この不透明ポインタは、当該ポインタが汎用的であるという既知の利点を提供し、その結果、不透明ポインタは、動的なポート番号を使用するあらゆる種類のネットワークサービス（たとえば、ＲＰＣサービスおよびＸＴＥＲＭサービス）をサポートすることができる。
ただし、各種のネットワークサービスに固有の他の種類のポイントを代わりに使用することもできる。
ポイント４２６のそれぞれは、関連するネットワークサービスのＭＡＣ規則を提供する、対応するアクセスチェックモジュール４２８を指し示す。
ＲＰＣネットワークサービスの場合、たとえば（図示するように）、モジュール４２８の名称は、そのサービスを表す「ＲＰＣ」であり、ポイント４２６は、ＲＰＣサービスのアクセスチェックモジュール４２８を指し示す。
既に示したように、このモジュール名は、「ＦＳ」（図３のファイル共有コンパートメントとすることができる）と呼ばれるコンパートメントに関連付けられている。
アクセスチェックモジュール４２８のこのインスタンスでは、（ＲＰＣプログラム番号によって識別される）ＲＰＣサービスへの（ＦＳコンパートメントのエンティティの）アクセスはＧＲＡＮＴである。
【００３４】
各アクセスチェックモジュールは、それぞれのネットワークサービスのＭＡＣ規則を提供する。
各ネットワークサービスが異なるとき、各アクセスチェックモジュールは、適切なそれぞれの機能を備え、通常は、異なる入力要件（すなわち、異なる入力引数）で異なって動作する傾向があるが、それぞれのネットワークサービスへのアクセスをＧＲＡＮＴまたはＤＥＮＹすることになる同じ種類のＭＡＣ出力を有する。
【００３５】
原則として、各種のアクセスチェックモジュールの多数の異なるインスタンスを有することが必要な場合がある。
異なるコンパートメントは、ＲＰＣサービス（または、実際には他の任意のサービス）の異なるアクセス制御要件を有する場合がある（また、おそらく有する）ので、たとえば、ＲＰＣアクセスチェックモジュールの複数のインスタンスが存在する場合がある。
たとえば、規則テーブルの第２のエントリー４３０は、「ＤＢ」コンパートメント（図３のデータベースコンパートメント３１５とすることができる）に関するものである。
ＤＢエントリー４３０のそれぞれのポインタ４３２は、第２のＲＰＣアクセスチェックモジュール４３４を識別する。
この第２のＲＰＣアクセスチェックモジュール４３４は、このインスタンスでは、識別されたＲＰＣサービスへのアクセスを拒否する。
実際には、いずれの特定のネットワークサービスについてもコンパートメント規則がない場合にも、アクセスは拒否されることになり、それによって、拒否されるコンパートメントおよびネットワークサービスのあらゆる組み合わせについてアクセスチェックモジュールが存在する必要があるという事態が回避される。
【００３６】
しかしながら、同じネットワークサービスが、異なるコンパートメントの要件に応じるために、複数のアクセスチェックモジュールを必要とする場合があるが、２つ以上のコン

パートメントが所与のネットワークサービスについて同じアクセス制御要件を有する場合に、コンパートメント規則テーブル４１８の２つのそれぞれのエントリーは、同じアクセスチェックモジュールを指し示すことができる。

【００３７】

単なる例示として、コンパートメント規則テーブル４１８の最後のエントリー４４０は、任意のネットワークサービス「ｘｙｚ」に関するものとして示されている。

このネットワークサービス「ｘｙｚ」は、動的に割り当てられるポートを使用する任意のネットワークサービスとすることができる。

それぞれのポインタ４４２は、ネットワークサービスへのアクセスを拒否する適切なアクセスチェックモジュール４４４を指し示す。

【００３８】

図５のフロー図を参照して、例示的なネットワークサービスＭＡＣプロセスを説明する。

この例では、ＲＰＣサービスコールを説明するが、まず、背景理解のためだけに、ＲＰＣサービスを一般的に説明する。

【００３９】

ＲＰＣによる例示的なリモートプロシージャコールは、コール側プロセスがコールメッセージをサーバプロセスへ送信して応答メッセージを待つことを伴う。

このコールメッセージは、プロシージャのパラメータを含み、応答メッセージは、プロシージャを実行したプロシージャ結果を含む。

コール側プロセスは、応答メッセージを受信すると、実行を続ける。

サーバ側では、ＲＰＣサービスは、通常、ドーマントであり、それぞれのＲＰＣコールメッセージの到着を待ち受けている。

ＲＰＣコールメッセージが到着すると、サービスは要求しているコール側プロセスへその後返信する応答を計算する。

【００４０】

一般に、ＲＰＣサービスは少なくとも、自身のＲＰＣプログラム番号、バージョン番号、および当該ＲＰＣサービスに到達することができるトランスポートアドレスによって識別することができる。

バージョン番号は、同じサービスのさまざまな異なるバージョン間を区別するために使用され、大部分は、時間と共にサービスを発展させるのに利用される。

トランスポートアドレスはネットワークアドレスおよびトランスポートセレクタから成る。

ＴＣＰまたはＵＤＰ上で利用可能なサービスの場合、ネットワークアドレスはＩＰアドレスであり、トランスポートセレクタはＴＣＰポート番号またはＵＤＰポート番号である。

コール側プロセスは、サービスを利用するために、そのサービスに対応するＲＰＣプログラム番号、バージョン番号、およびトランスポートアドレスを知っている必要がある。

これらのうち、ＲＰＣプログラム番号およびバージョン番号は、サービス定義の一部としてコール側プロセスに組み込むことができる。

トランスポートアドレスのネットワークアドレスコンポーネントは、通例、（たとえば、ＨＰ－ＵＸのディレクトリパス/usr/sbin/namedに見られる）「named」等のネームサービスデーモンから取得可能であるか、またはコール側プロセスにパラメータとして提供することができる。

本明細書で既に説明したように、ＲＰＣサービスの場合、トランスポートセレクタ（すなわち、ＴＣＰポートまたはＵＤＰポート）は通例、動的に決定され、サービスの各起動と共に変化する。

サーバプログラムは、トランスポートアドレスを配分し、ルックアップサービス（rpcbindまたはbind等）に登録する。

ルックアップサービスは、固定されたトランスポートセレクタ（すなわち、ＴＣＰポー

ト番号１１１）に関連付けられ、サーバと同じネットワークアドレスに存在する。
コール側プロセスは通常、サーバのトランスポートアドレスを取得するために、ルックアップサービスを調べることが可能であり、その後、ＲＰＣサービスを起動して完了することができる。

【００４１】

次に図５に戻って、最初のステップ５００において、システムは、たとえばypserv ＲＰＣサービスのＲＰＣコールを受信する。
このＲＰＣコールは、ypservのＲＰＣプログラム番号を含めて受信される。
プログラム番号は１００００４である。
一般に既知のシステムでは、ＲＰＣコールはマシンにおいて既知のＲＰＣサービス（またはプロセス）によって受信される。
この既知のＲＰＣサービス（またはプロセス）は、rpc_call()と呼ばれ、rpc_call()は、ＲＰＣプログラム番号およびバージョン番号を含む、多数の引数またはパラメータを受信する予定である。
本発明の実施形態によれば、ＲＰＣコールは、システムコールインターフェース４０６によって受信されると、ネットワークサービスコールとして認識され、ネットワークサービスハンドラ４０８に渡される。
ネットワークサービスハンドラ４０８は、rpc_call()プロセスの修正バージョンを組み込む。
このrpc_call()プロセスは、たとえば、以下の形を有する新しいセキュリティコンテインメント関数コールを含めることによって適合されている。

```
"network_services_mac_access_check(
      network_service_name,
      rcp_info_pointer,
      compartment)."
```

【００４２】

本発明のypservの例によれば、network_services_mac_access_check()の引数は、

```
network_service_name="RPC"
rcp_info_pointer=100004
compartment=FS
```

である。

【００４３】

ＸＴＥＲＭ等の他のサービスの場合、rpc_call()プロセスの均等物は、network_services_mac_access_check()関数コール（すなわち「hook」）を含むようになっている。
ただし、もちろん、パラメータは、ＲＰＣサービスのパラメータではなくＸＴＥＲＭサービスに必要とされるパラメータに一致する。
network_services_mac_access_check ()関数自体は、セキュリティコンテインメントコントローラの一部である。

【００４４】

図５の次のステップ５０５において、network_services_mac_access_check()関数は、ＦＳコンパートメントおよびモジュール名「ＲＰＣ」の双方に対応するエントリーを識別するために、セキュリティコンテインメント規則関数４１２を検索する。
ステップ５１０において、エントリーが存在しない場合には、デフォルトの応答は、アクセス制御要求がステップ５１５において拒否されるというものであり、この応答は、network_services_mac_access_check()関数を介してrpc_call()プロセスに戻され、コール側エンティティには、従来の方法で、アクセスが拒否されることが通知される。
一方、エントリーが存在する場合には（図４によれば、エントリーは存在する）、ステップ５２０において、それぞれの不透明ポインタ４２６が使用されて、対応するアクセスチェックモジュール４２８が識別される。
ステップ５２５において、モジュールが存在しない場合には、この場合も、デフォルト

の応答は、アクセス要求がステップ５１５において拒否されるというものである。
モジュールが存在する場合（図４では、「ＲＰＣアクセスチェックモジュール」という名称のものが存在する）には、ステップ５３０において、そのモジュールは、rpc_info_pointer値を使用して、ypservプロシージャのそれぞれのアクセス許可を識別する。
図４の図解の複雑さを低減するために図示されていないが、ＲＰＣアクセスチェックモジュール４３４は、１つ以上のＲＰＣネットワークサービスのアクセス制御規則と、（ＲＰＣプログラム番号によって識別される）指定されたサービスの適切なアクセス制御規則を検索するためのロジックとを定義するテーブルを含む。
ステップ５３５において、アクセス許可が存在し、且つ、ＧＲＡＮＴである場合には、ステップ５４０において、サービスへのアクセスが付与され、許可は、network_services_mac_access_check()関数を介してrpc_call()プロセスへ戻される。
rpc_call()プロセスは、通常の方法によるターゲットＲＰＣサーバとの相互作用でＲＰＣサービスを起動することを引き受ける。
アクセス許可が「ＤＥＮＹ」である（または、指定されたＲＰＣネットワークサービスについて設定されたアクセス許可が存在しない）場合には、アクセス許可は、ステップ５１５において拒否され、応答は、network_services_mac_access_check()関数を介してrpc_call()プロセスへ戻され、コール側エンティティには、従来の方法でアクセスが拒否されことが通知される。

【００４５】

ネットワークサービスアクセス許可は、本発明の実施形態によれば、適切なエントリーをセキュリティコンテインメント規則関数４１２に追加することによって設定される。
セキュリティコンテインメント規則関数４１２は、図４の図解で示すように、セキュリティコンテインメントコントローラ４１０によって使用される。

【００４６】

次に、ＨＰ-ＵＸ　Ｓｅｃｕｒｉｔｙ　Ｃｏｎｔａｉｎｍｅｎｔセキュアオペレーティングシステムにおいて、本発明の実施形態に従ってネットワークサービスアクセス許可を設定する一例を、「ＦＳ」と呼ばれる例示のコンパートメントについて説明する。
アクセス許可は、他のコンパートメント規則と同様に、database.rulesと呼ばれる（ＨＰ－ＵＸ製品内の）テキストファイルに入力される。
このテキストファイルは以下の形を取る。

```
compartment FS {
    DPR G RPC 100003 {grant access to 'NFS'}
    DPR G RPC 100004 {grant access to 'ypserv'}
    DPR G RPC 100005 {grant access to 'mountd'}
    DPR G RPC 100006 {grant access to 'rpcbind'}
    DPR G XTERM putty {grant access to 1st xterm service, 'putty'}
    DPR D XTERM refx {deny access to 2nd xterm service, 'refx'}
}
```

【００４７】

上記は、「ＦＳ」と呼ばれるコンパートメントを定義し、第１のＸＴＥＲＭサービス「putty」へのアクセスを付与すること、および、第２のＸＴＥＲＭサービス「refx」へのアクセスを拒否することに加えて、ＲＰＣサービス「NFS」、「ypserv」、および「mountd」へのアクセスを付与するＭＡＣ規則も定義している。

【００４８】

同様のテキスト構造は、ＭＡＣ規則が必要とされる他のあらゆるコンパートメントについても提供される。

【００４９】

説明してきたように、本発明の実施形態に従って動作するように適合することができるＨＰ－ＵＸ　Ｓｅｃｕｒｉｔｙ　Ｃｏｎｔａｉｎｍｅｎｔオペレーティングシステムによれば、コンパートメントおよびそれぞれのＭＡＣ規則は通常、システムディレクトリ/usr

/sbinに見られる、setrules（規則設定）と呼ばれるプログラムを使用して、セキュリティコンテインメント規則関数４１２内にロードされる。
setrulesプログラムは、database.rulesを解析すると共に、コンパートメントの各規則を順に設定するように構成されている。
setrulesプログラムは、アドミニストレータがdatabase.rulesを記憶しなければならない場所である/etc/cmptと呼ばれるシステムディレクトリからdatabase.rulesを読み出すように構成されている。
【００５０】
システムアドミニストレータは、以下のコマンドラインを使用してsetrulesを実行する。
/usr/sbin/setrules
【００５１】
図４に示すセキュアオペレーティングシステムは、setrulesプログラムを表す機能ブロック４５０を含む。
既に説明したように、このプログラムは指定されたディレクトリ/usr/sbin/setrulesに存在するが、インターフェースを提供するので、セキュリティコンテインメントコントローラ４１０の一部として示されている。
このインターフェースによって、システムアドミニストレータはカーネルコードおよび関連する定義にアクセスすると共にそれらを生成または修正する必要なく、テキストファイルの定義に基づきコンパートメントをセットアップすることが可能になる。
システムアドミニストレータは、上述したようにコンパートメントのＭＡＣポリシーを定義することによってコンパートメントをセットアップしたいとき、たとえばdatabase.rulesと呼ばれるテキストファイルを生成し、そのテキストファイルを/etc/cmptディレクトリに記憶する。
次に、システムアドミニストレータは、
/usr/sbin/setrules
等のコマンドラインを発行して、コンパートメントの生成を開始する。
システムコールインターフェース４０６は、このコマンド入力を受信して認識し、アクセス設定ハンドラ（Set Access Handler）４５２へ転送する。
アクセス設定ハンドラ４５２は、次に、setrules関数にアクセスし、コンパートメント定義をロードする。
【００５２】
次に、図６のフロー図を参照して、動的に割り当てられるポート番号を使用するネットワークサービスにアクセスするためのＭＡＣ規則を構築するための例示的なプロセスを説明する。
図６のフロー図は、ＦＳコンパートメントについて上述したコンパートメント定義に基づいている。
【００５３】
図６の最初のステップ６００において、システムコールインターフェース４０６は、setrulesコマンドを受信し、そのコマンドをアクセス設定ハンドラ４５２に渡す。
アクセス設定ハンドラ４５２は、ステップ６０５において、setrulesプログラムを起動する。
setrulesプログラムは、ステップ６１０において、database.rulesファイルにアクセスし、定義テキストの最初のラインを解析する。
この例では、定義ファイルの最初のラインは、「ＦＳ」と呼ばれる新しいコンパートメントを定義する。
ステップ６１５において、ラインがコンパートメント定義ラインである場合には、ステップ６２０において、新しいコンパートメントエントリーがセキュアオペレーティングシステムのコンパートメントリスト（図示せず）に追加され、後続のあらゆるＭＡＣ規則がそのコンパートメントに関連付けられ、プロセスはステップ６１０に戻る。

ステップ６１０において、定義テキストの次のラインが解析される。
ステップ６１５において、ラインがコンパートメントのＤＰＲ規則である場合には、プロセスはステップ６２５に続く。
この例では、ＤＰＲ規則のプロセスしか含まれていないが、他の既知のプロシージャおよび同等のプロシージャがこのような他の規則の種類について提供されることが認識されよう。
ステップ６２５において、プロセスは（<MODULE>フィールドを使用して）、規則が既存のアクセスチェックモジュールへの追加であるか、または、規則が新しいアクセスチェックモジュールの生成を必要としているかを識別する。
規則が新しいアクセスチェックモジュールに関するものである場合、ステップ６３０において、このプロセスは、それぞれのアクセスチェックモジュールの適切なＲＰＣサービステンプレートを見つけ（ＲＰＣサービステンプレートでは、規則を設定することができる既知の各サービスが、特定のアクセスチェックモジュールのフレームワークを提供する事前に定義されているテンプレートを有する）、（識別子としての<MODULE>フィールド、ＲＰＣプログラム番号フィールドを満たす<STRING>値１０００００３、および許可であるＧ／Ｄ値を使用して）新しいアクセスチェックモジュールを生成する。
ステップ６３５において、このプロセスはエントリーを規則テーブル４１８に追加する。
このエントリーは、コンパートメントＩＤを含む第１のフィールド４２０、モジュール名を含む第２のフィールド４２２、および（第３のフィールド４２４の）不透明ポインタを含む。
この不透明ポインタはテーブルエントリーをアクセスチェックモジュールコードに向けるものである。
次に、ステップ６４０において、このプロセスは規則が定義内にさらに存在するか否かを判断する。
定義する規則がさらに存在する場合には、このプロセスはステップ６１０に戻って、定義の次のラインを解析する。
定義する規則がそれ以上存在しない場合には、このプロセスはステップ６４５において終了する。

【００５４】

ステップ６２５において、新しい規則が既存のアクセスチェックモジュールに追加されると判断された場合には、ステップ６５０において、このプロセスは、ＲＰＣプログラム番号フィールドを満たす<STRING>値１０００００３、および許可であるＧ／Ｄ値を使用して、既存のアクセスチェックモジュールにエントリーを追加する。
このプロセスは次に、ステップ６４０に続く。

【００５５】

図６のプロセスは、新しい規則が発見されなくなるまで定義内の規則ごとに繰り返され、そして、プロセスは終了する。
その結果、規則テーブルは、セキュアオペレーティングシステムによってネットワークサービスへのアクセスを制御するのに必要とされるすべてのコンパートメント規則への参照子でポピュレートされる。

【００５６】

上記実施形態は本発明の説明例として理解されるべきである。
本発明のさらなる実施形態が予想される。
ＨＰ－ＵＸ Ｓｅｃｕｒｉｔｙ Ｃｏｎｔａｉｎｍｅｎｔセキュアオペレーティングシステムを適合させることによって本発明の特定の実施形態を実施できることが示されたが、ＳＥＬｉｎｕｘ（商標）またはＴｒｕｓｔｅｄ Ｓｏｌａｒｉｓ（商標）等の他の既知のセキュアオペレーティングシステム、または実際には、動的に割り当てられるポートを使用するネットワークサービスにアクセスするためのコンパートメント規則を定義するように適合することができる他の任意のセキュアオペレーティングシステムを適合させるこ

とによって、他の実施形態を実施することができる。

【００５７】

本発明の実施形態は、ハードウェア、ソフトウェア、またはハードウェアおよびソフトウェアの組み合わせの形で実現することができることが認識されよう。

このようなどのソフトウェアも、たとえば、消去可能であろうとなかろうと、再書き込み可能であろうとなかろうと、ＲＯＭのようなストレージデバイス等の揮発性ストレージまたは不揮発性ストレージの形で記憶することもできるし、たとえば、ＲＡＭ、メモリチップ、デバイス、または集積回路等のメモリの形で記憶することもできるし、たとえば、ＣＤ、ＤＶＤ、または磁気ディスク、磁気テープ等の光可読媒体または磁気可読媒体に記憶することもできる。

これらのストレージデバイスおよびストレージ媒体は、実行されると本発明の実施形態を実施する１つ以上のプログラムを記憶するのに適したマシン可読ストレージの実施形態であることが認識されよう。

したがって、実施形態は、先行するいずれかの請求項に記載のシステムまたは方法を実施するためのコードを含むプログラム、および、このようなプログラムを記憶する機械可読ストレージを提供する。

またさらに、本発明の実施形態は、有線接続または無線接続によって運ばれる通信信号等のあらゆる媒体を介して電子的に搬送することもでき、実施形態はそれらも適切に包含する。

【００５８】

本明細書（添付のあらゆる請求項、要約書、および図面を含む）に開示された機能のすべて、および／または、そのように開示されたあらゆる方法若しくはプロセスのステップのすべては、このような機能および／またはステップの少なくともいくつかが相互に排他的である組み合わせを除いて、任意の組み合わせで組み合わせることができる。

【００５９】

本明細書（添付のあらゆる請求項、要約書、および図面を含む）に開示した各機能は、明示的に別段の定めをした場合を除き、同じ目的、等価な目的、または類似の目的を果たす代替的な機能に置き換えることができる。

したがって、明示的に別段の定めをした場合を除き、開示された各機能は、一般的な一連の等価な機能または類似の機能の一例にすぎない。

【００６０】

本発明は、上記のいずれの実施形態の詳細にも限定されるものではない。

本発明は、本明細書（添付したあらゆる請求項、要約書、および図面を含む）に開示した機能の新規ないずれのものにも若しくは新規ないずれの組み合わせにもおよび、または、そのように開示されたあらゆる方法またはプロセスのステップの新規ないずれのものにも若しくは新規ないずれの組み合わせにも及ぶ。

特許請求の範囲は、上記実施形態のみを包含するように解釈されるべきではなく、特許請求の範囲の範中に含まれるあらゆる実施形態も包含するように解釈されるべきである。

【図面の簡単な説明】

【００６１】

【図１】本発明の実施形態によるオペレーションに適した例示的なコンピューティング環境を示す図解である。

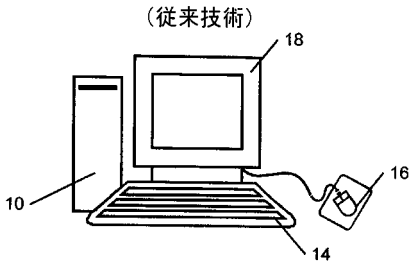
【図２】図１に示す種類のコンピューティングプラットフォームと共に使用するための例示的なマザーボードを示す図解である。

【図３】分散コンピューティング環境における３つのコンピューティングプラットフォームを示す概略図解である。第１のプラットフォームはＲＰＣサーバとして動作し、第２のプラットフォームはコンパートメント化されたオペレーティングシステムを実行するクライアントとして動作し、第３のプラットフォームはウェブサーバとして動作する。

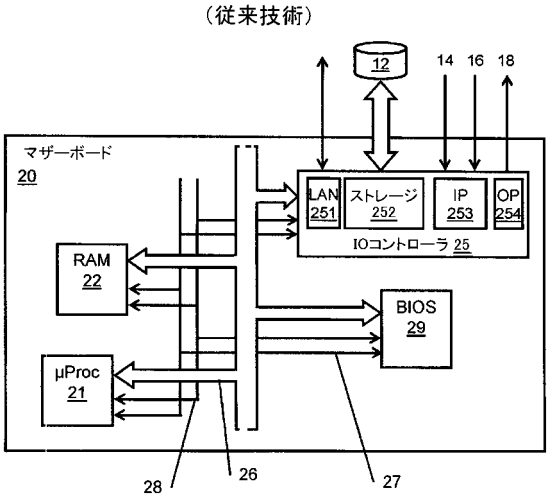
【図４】本発明の一実施形態を実施するのに必要なセキュアオペレーティングシステムの機能要素を示す概略図解である。

【図５】本発明の一実施形態によるネットワークサービスへのアクセスを制御することに関与するステップを示すフロー図である。
【図６】動的なポート割り当てを使用するネットワークサービスの新しいコンパートメント強制アクセス制御規則を定義することに関与するステップを示すフロー図である。
【符号の説明】
【００６２】
３００・・・マシン１，
４０２・・・カーネル空間，
４０４・・・ユーザ空間，
４０６・・・システムコールインターフェース，
４０８・・・ネットワークサービスハンドラ，
４１０・・・セキュリティコンテインメントコントローラ，
４１２・・・セキュリティコンテインメント規則，
４１４・・・ＬＡＮカードコントローラ，
４１６・・・ＬＡＮカード，
４２０・・・コンパートメントＩＤ，
４２２・・・モジュール名，
４２４・・・ポインタ，
４２８・・・ＲＰＣアクセスチェックモジュール，
４５２・・・アクセス設定ハンドラ，

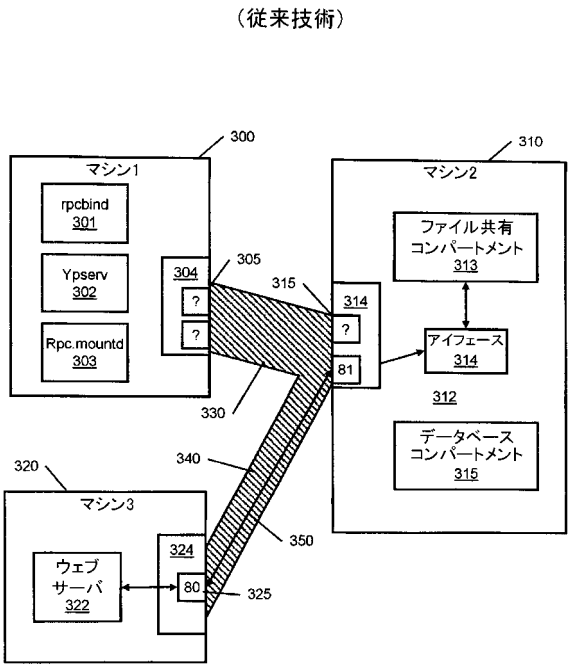
【図１】



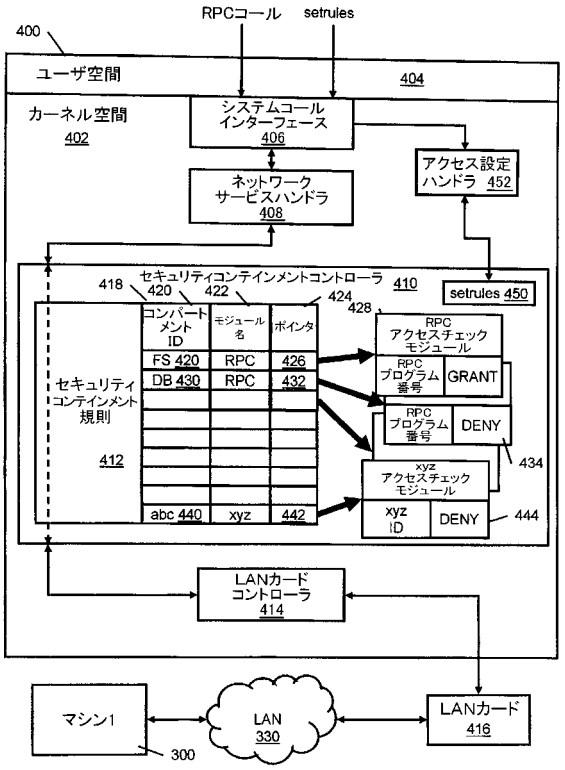
【図２】



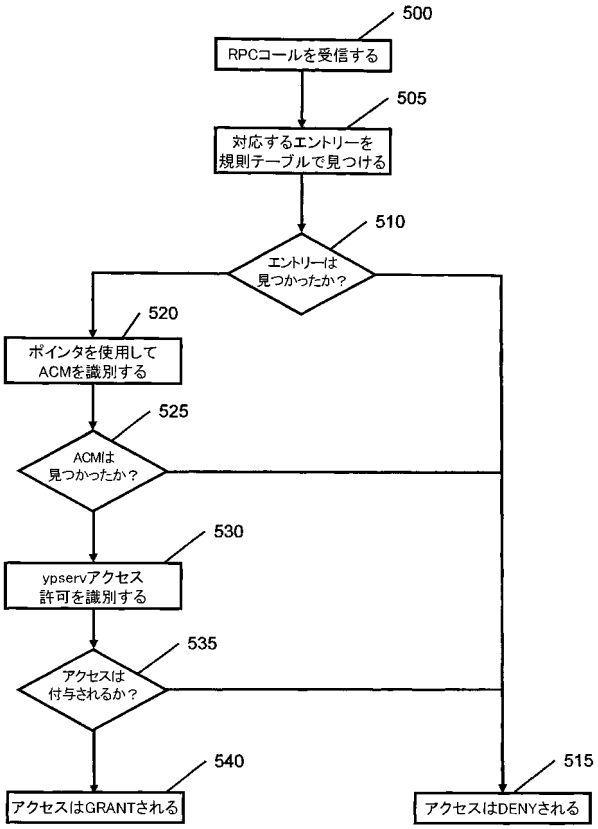
【図３】



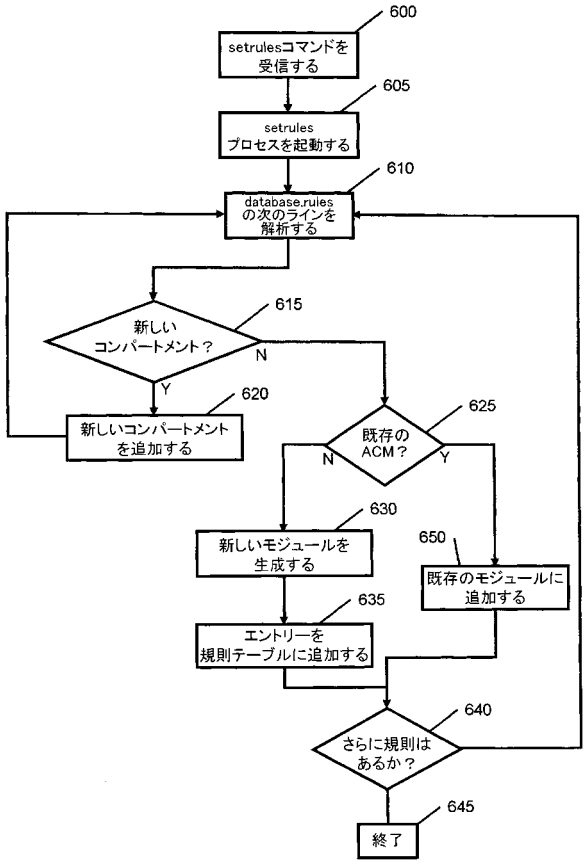
【図４】


400
RPCコール
setrules
ユーザ空間
404
カーネル空間
402
システムコール
インターフェース
406
アクセス設定
ハンドラ 452
ネットワーク
サービスハンドラ
408
セキュリティコンテインメントコントローラ 410
setrules 450
418
420
422
424
428
コンパートメント ID
モジュール名
ポインタ
FS 420
RPC
426
DB 430
RPC
432
セキュリティコンテインメント規則
412
abc 440
xyz
442
RPC アクセスチェックモジュール
RPC プログラム番号
GRANT
RPC プログラム番号
DENY
434
xyz アクセスチェックモジュール
xyz ID
DENY
444
LANカード コントローラ
414
マシン1
300
LAN
330
LANカード
416

【図５】


500
RPCコールを受信する
505
対応するエントリーを規則テーブルで見つける
510
エントリーは見つかったか？
520
ポインタを使用してACMを識別する
525
ACMは見つかったか？
530
ypservアクセス許可を識別する
535
アクセスは付与されるか？
540
アクセスはGRANTされる
515
アクセスはDENYされる

【図６】


600
setrulesコマンドを受信する
605
setrules プロセスを起動する
610
database.rules の次のラインを解析する
615
新しいコンパートメント？
N
Y
620
新しいコンパートメントを追加する
625
既存のACM？
N
Y
630
新しいモジュールを生成する
650
既存のモジュールに追加する
635
エントリーを規則テーブルに追加する
640
さらに規則はあるか？
645
終了

フロントページの続き

(72)発明者　ムラリ・サブラマニアン
インド国カルナタカ　バンガロール　マハデバプラポスト　ホワイトフィールドロード　エヌオー　１９２　エスワイ　ヒューレット・パッカード・インディア・ソフトウェア・オペレーション・プライベート・リミティッド内
(72)発明者　アナンサ・キールシー・バナバラ・ラマスワミー
インド国カルナタカ　バンガロール　マハデバプラポスト　ホワイトフィールドロード　エヌオー　１９２　エスワイ　ヒューレット・パッカード・インディア・ソフトウェア・オペレーション・プライベート・リミティッド内
(72)発明者　アラン・ケシャバ・マーシー
インド国カルナタカ　バンガロール　マハデバプラポスト　ホワイトフィールドロード　エヌオー　１９２　エスワイ　ヒューレット・パッカード・インディア・ソフトウェア・オペレーション・プライベート・リミティッド内

審査官　和田　財太

(56)参考文献　三田聖彦, マルチプラットフォームが特徴のセキュアＯＳ・トラステッドＯＳ　ＰｉｔＢｕｌｌ, ＵＮＩＸ　ｍａｇａｚｉｎｅ, 日本, 株式会社アスキー, ２００７年　４月　１日, Ｖｏｌ．２２　Ｎｏ．２, ｐ．９２－ｐ．９９

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ　２１／００－２１／２４
Ｇ０６Ｆ　１３／００