(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7546602号
(P7546602)

(45)発行日　令和6年9月6日(2024.9.6)

(24)登録日　令和6年8月29日(2024.8.29)

(51)国際特許分類

Ｇ０６Ｆ　21/55　(2013.01)		Ｇ０６Ｆ	21/55	
Ｇ０６Ｆ　12/14　(2006.01)		Ｇ０６Ｆ	12/14	５１０Ｄ

請求項の数　15　（全14頁）

(21)出願番号　特願2021-568575(P2021-568575)
(86)(22)出願日　令和2年5月15日(2020.5.15)
(65)公表番号　特表2022-534685(P2022-534685A)
(43)公表日　令和4年8月3日(2022.8.3)
(86)国際出願番号　PCT/EP2020/063618
(87)国際公開番号　WO2020/234155
(87)国際公開日　令和2年11月26日(2020.11.26)
審査請求日　令和4年10月21日(2022.10.21)
(31)優先権主張番号　16/416,229
(32)優先日　令和1年5月19日(2019.5.19)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国１０５０４　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
Ｎｅｗ　Ｏｒｃｈａｒｄ　Ｒｏａｄ，　Ａｒｍｏｎｋ，　Ｎｅｗ　Ｙｏｒｋ　１０５０４，　Ｕｎｉｔｅｄ　Ｓｔａｔｅｓ　ｏｆ　Ａｍｅｒｉｃａ
(74)代理人　100112690
弁理士　太佐　種一

最終頁に続く

(54)【発明の名称】　オペレーティング・システム・カーネルの分離されたアドレス空間におけるシステム・コールの実行

(57)【特許請求の範囲】

【請求項１】

コンピュータが実行する方法であって、

テンプレート・システム・コール・ページ・テーブルを作成することであって、

システム・コールの実行時にカーネル・モードへのエントリを実行するために必要なカーネル・コードおよびカーネル・データと、システム・コール・ハンドラへのポインタを有するテーブルと、各システム・コール・ハンドラのエントリ・コードとのマッピングと、

カーネル・エントリ・ページ・テーブルと、

空のユーザ空間ページ・テーブルと、を含む、

前記テンプレート・システム・コール・ページ・テーブルを作成することと、

ユーザ空間プロセスが作成されることに応答して、前記テンプレート・システム・コール・ページ・テーブルをプロセス状態表現にコピーして、前記ユーザ空間プロセスについてのオペレーション・システム・コール・ページ・テーブルを生成し、前記オペレーション・システム・コール・ページ・テーブルの前記空のユーザ空間ページ・テーブルに、ユーザ空間マッピングを入力することと、

前記ユーザ空間プロセスによって前記システム・コールを実行することと、

前記カーネル・モードに移行し、汎用システム・コール・エントリ・ルーチンを呼び出すことであって、前記汎用システム・コール・エントリ・ルーチンは、前記オペレーション・システム・コール・ページ・テーブルを使用するため、および前記オペレーション・システム・コール・ページ・テーブルの前記システム・コール・ハンドラを実行するため

に、ページ・テーブル・ポインタ・レジスタを切り替える、呼び出すことと、
前記システム・コール・ハンドラがカーネル空間の前記カーネル・コードまたは前記カーネル・データあるいはその両方を含むメモリへのアクセスを試行することに応答して、ページ・フォールトを発生させることと、
前記メモリへの前記アクセスが許可されているかどうかを判定することと、
前記アクセスが許可されているという判定に応答して、前記メモリのマッピングを前記オペレーション・システム・コール・ページ・テーブルに追加することと、
前記システム・コール・ハンドラの実行が終了することに応答して、制御を汎用システム・コール・エントリ・ルーチンに戻すことと、
を含む、方法。

【請求項２】
前記コンピュータのオペレーティング・システムがＬｉｎｕｘオペレーティング・システムである、請求項１に記載の方法。

【請求項３】
前記オペレーション・システム・コール・ページ・テーブルが、カーネル・コードの実行およびカーネル・データへのアクセスを許可する複数のカーネル空間アドレス範囲を含む、請求項１または２に記載の方法。

【請求項４】
前記アクセスが許可されていないと判定されたことに応答して、前記カーネル空間の前記メモリへのアクセスを試行した前記システム・コールを行った前記ユーザ空間プロセスの実行を終了することをさらに含む、請求項１から請求項３のいずれか一項に記載の方法。

【請求項５】
前記システム・コール・ハンドラの実行が終了することに応答して、前記システム・コール・ハンドラの実行中に前記オペレーション・システム・コール・ページ・テーブルに追加された前記マッピングを除去することをさらに含む、請求項１から請求項４のいずれか一項に記載の方法。

【請求項６】
プロセッサと、前記プロセッサによってアクセス可能なメモリと、前記メモリに格納されたコンピュータ・プログラム命令とを含むシステムであって、
前記コンピュータ・プログラム命令が、
テンプレート・システム・コール・ページ・テーブルを作成することであって、
システム・コールの実行時にカーネル・モードへのエントリを実行するために必要なカーネル・コードおよびカーネル・データと、システム・コール・ハンドラへのポインタを有するテーブルと、各システム・コール・ハンドラのエントリ・コードとのマッピングと、カーネル・エントリ・ページ・テーブルと、
空のユーザ空間ページ・テーブルと、を含む、
前記テンプレート・システム・コール・ページ・テーブルを作成することと、
ユーザ空間プロセスが作成されることに応答して、前記テンプレート・システム・コール・ページ・テーブルをプロセス状態表現にコピーして、前記ユーザ空間プロセスについてのオペレーション・システム・コール・ページ・テーブルを生成し、前記オペレーション・システム・コール・ページ・テーブルの前記空のユーザ空間ページ・テーブルに、ユーザ空間マッピングを入力することと、
前記ユーザ空間プロセスによって前記システム・コールを実行することと、
前記カーネル・モードに移行し、汎用システム・コール・エントリ・ルーチンを呼び出すことであって、前記汎用システム・コール・エントリ・ルーチンは、前記オペレーション・システム・コール・ページ・テーブルを使用するため、および前記オペレーション・システム・コール・ページ・テーブルの前記システム・コール・ハンドラを実行するために、ページ・テーブル・ポインタ・レジスタを切り替える、呼び出すことと、
前記システム・コール・ハンドラがカーネル空間の前記カーネル・コードまたは前記カーネル・データあるいはその両方を含むメモリへのアクセスを試行することに応答して、

ページ・フォールトを発生させることと、
前記メモリへの前記アクセスが許可されているかどうかを判定することと、
前記アクセスが許可されているという判定に応答して、前記メモリのマッピングを前記オペレーション・システム・コール・ページ・テーブルに追加することと、
前記システム・コール・ハンドラの実行が終了することに応答して、制御を汎用システム・コール・エントリ・ルーチンに戻すことと、
を実行するように、前記プロセッサによって実行可能である、システム。
【請求項７】
前記システムのオペレーティング・システムがＬｉｎｕｘオペレーティング・システムである、請求項６に記載のシステム。
【請求項８】
前記オペレーション・システム・コール・ページ・テーブルが、カーネル・コードの実行およびカーネル・データへのアクセスを許可する複数のカーネル空間アドレス範囲を含む、請求項６または請求項７に記載のシステム。
【請求項９】
前記コンピュータ・プログラム命令が、前記アクセスが許可されていないと判定されたことに応答して、前記カーネル空間の前記メモリへのアクセスを試行した前記システム・コールを行った前記ユーザ空間プロセスの実行を終了することをさらに実行するように、前記プロセッサによって実行可能である、請求項６から請求項８のいずれか一項に記載のシステム。
【請求項１０】
前記コンピュータ・プログラム命令が、前記システム・コール・ハンドラの実行が終了することに応答して、前記システム・コール・ハンドラの実行中に前記オペレーション・システム・コール・ページ・テーブルに追加された前記マッピングを除去することをさらに実行するように、前記プロセッサによって実行可能である、請求項６から請求項９のいずれか一項に記載のシステム。
【請求項１１】
テンプレート・システム・コール・ページ・テーブルを作成することであって、
システム・コールの実行時にカーネル・モードへのエントリを実行するために必要なカーネル・コードおよびカーネル・データと、システム・コール・ハンドラへのポインタを有するテーブルと、各システム・コール・ハンドラのエントリ・コードとのマッピングと、
カーネル・エントリ・ページ・テーブルと、
空のユーザ空間ページ・テーブルと、を含む、
前記テンプレート・システム・コール・ページ・テーブルを作成することと、
ユーザ空間プロセスが作成されることに応答して、前記テンプレート・システム・コール・ページ・テーブルをプロセス状態表現にコピーして、前記ユーザ空間プロセスについてのオペレーション・システム・コール・ページ・テーブルを生成し、前記オペレーション・システム・コール・ページ・テーブルの前記空のユーザ空間ページ・テーブルに、ユーザ空間マッピングを入力することと、
前記ユーザ空間プロセスによって前記システム・コールを実行することと、
前記カーネル・モードに移行し、汎用システム・コール・エントリ・ルーチンを呼び出すことであって、前記汎用システム・コール・エントリ・ルーチンは、前記オペレーション・システム・コール・ページ・テーブルを使用するため、および前記オペレーション・システム・コール・ページ・テーブルの前記システム・コール・ハンドラを実行するために、ページ・テーブル・ポインタ・レジスタを切り替える、呼び出すことと、
前記システム・コール・ハンドラがカーネル空間の前記カーネル・コードまたは前記カーネル・データあるいはその両方を含むメモリへのアクセスを試行することに応答して、
ページ・フォールトを発生させることと、
前記メモリへの前記アクセスが許可されているかどうかを判定することと、
前記アクセスが許可されているという判定に応答して、前記メモリのマッピングを前記

オペレーション・システム・コール・ページ・テーブルに追加することと、
前記システム・コール・ハンドラの実行が終了することに応答して、制御を汎用システム・コール・エントリ・ルーチンに戻すことと、
をコンピュータに実行させる、コンピュータ・プログラム。
【請求項１２】
前記コンピュータのオペレーティング・システムがＬｉｎｕｘオペレーティング・システムである、請求項１１に記載のコンピュータ・プログラム。
【請求項１３】
前記オペレーション・システム・コール・ページ・テーブルが、カーネル・コードの実行およびカーネル・データへのアクセスを許可する複数のカーネル空間アドレス範囲を含む、請求項１１または請求項１２に記載のコンピュータ・プログラム。
【請求項１４】
前記アクセスが許可されていないと判定されたことに応答して、前記カーネル空間の前記メモリへのアクセスを試行した前記システム・コールを行った前記ユーザ空間プロセスの実行を終了することをさらにコンピュータに実行させる、請求項１１から請求項１３のいずれか一項に記載のコンピュータ・プログラム。
【請求項１５】
前記システム・コール・ハンドラの実行が終了することに応答して、前記システム・コール・ハンドラの実行中に前記オペレーション・システム・コール・ページ・テーブルに追加された前記マッピングを除去することをさらにコンピュータに実行させる、請求項１１から請求項１４のいずれか一項に記載のコンピュータ・プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、専用アドレス空間でシステム・コールを実行することにより、オペレーティング・システム・カーネル自体の内部に追加のセキュリティ機構を提供して、悪意のあるアプリケーションに可視であるためこのアプリケーションによって悪用可能な共有リソースの量を削減する技術に関する。
【背景技術】
【０００２】
システム・コールは、ユーザ空間アプリケーションに公開される、Ｌｉｎｕｘなどのオペレーティング・システムのカーネル・アプリケーション・バイナリ・インタフェース（ＡＢＩ）の重要な部分である。システム・コールが実行されると、ＣＰＵは非特権モード（ｘ８６のＲｉｎｇ－３など）からスーパユーザ・モード（ｘ８６のＲｉｎｇ－０など）に移行し、システム・コールの実行中に呼び出されるルーチンは、最高の特権レベルを有するため、システム・リソースにアクセスしてこれを変更することができる。悪意のあるユーザ空間アプリケーションは、システム・コール実装の脆弱性を悪用して、これらのシステム・リソースにアクセスし、システムをクラッシュさせ、機密情報を読み取り、またはスーパユーザ特権を得ることがある。
【０００３】
マイクロカーネルは、スーパユーザ・モードで実行されるコードの量を最小限にし、分離されたアドレス空間でユーザ権限を用いてオペレーティング・システム・サービスのほとんどを実行することによって、特権状態の広範囲にわたる公開を回避する。マイクロカーネル手法の主な欠点は、プロセス間通信に関連するパフォーマンス・コストと、システムの様々なコンポーネント間のインタフェースの複雑さである。
【０００４】
メルトダウンの脆弱性の軽減として、ユーザとカーネル・アドレス空間とを完全に分離するページ・テーブル分離（ＰＴＩ）機構が、Ｌｉｎｕｘカーネルに追加された。しかしながら、ＰＴＩを使用しても、カーネル・コード全体が同一のアドレス空間を共有し、いずれかのシステム・コール・ハンドラにおける悪用によってシステム全体が脆弱になる。

【０００５】
したがって、オペレーティング・システム・カーネル自体の内部に追加のセキュリティ機構を提供する技術に対する必要性が生じる。
【発明の概要】
【０００６】
本システムおよび方法の実施形態は、専用アドレス空間でシステム・コールを実行することにより、オペレーティング・システム・カーネル自体の内部に追加のセキュリティ機構を提供して、悪意のあるアプリケーションに可視であるためこのアプリケーションによって悪用可能な共有リソースの量を削減することができる。
【０００７】
実施形態において、ユーザ空間アプリケーションがシステム・コールを実行するときに、カーネル・エントリ・コードは、その特定のシステム・コールの実行に適したアドレス空間を選択することができる。システム・コールの実行が、そのアドレス空間の外部のメモリへのアクセスを試行すると、ページ・フォールトが発生する。ページ・フォールト・ハンドラは、要求されたメモリ範囲が、システム・コール・ハンドラ・ルーチンによってアクセス許可されているかどうかを検証し、アクセス許可されていれば、この範囲を専用アドレス空間に追加することができる。システム・コールの完了時に、アドレス空間定義を元の状態に戻すことができる。
【０００８】
例えば、実施形態において、プロセッサと、プロセッサによってアクセス可能なメモリと、メモリに格納され、プロセッサによって実行可能なコンピュータ・プログラム命令とを含み得るコンピュータに実装された方法は、ユーザ・プロセスがシステム・コールを行うときに、カーネル・モードに切り替え、ユーザ・プロセスのためのシステム・コール・ページ・テーブルを使用してシステム・コール・ハンドラを実行することと、システム・コール・ハンドラが、マップされていないカーネル空間メモリへのアクセスを試行するときに、ページ・フォールトを発生させることと、マップされていないカーネル空間メモリに対して試行されたアクセスが許可されているかどうかを判定することによって、ページ・フォールトを処理することとを含むことができる。
【０００９】
実施形態において、コンピュータ・システムのオペレーティング・システムはＬｉｎｕｘオペレーティング・システムであってよい。システム・コール・ページ・テーブルは、カーネル・コードの実行およびカーネル・データへのアクセスを許可する複数のカーネル空間アドレス範囲を含むことができる。システム・コール・ページ・テーブルは、システム・コールの実行時にカーネル・モードへのエントリを実行するために必要なカーネル・コードおよびデータ、システム・コール・ハンドラへのポインタを有するテーブル、ならびに各システム・コール・ハンドラのエントリ・コードのマッピングを含むテンプレート・システム・コール・ページ・テーブルを作成することと、ユーザ・プロセスが作成されたときに、ユーザ・プロセスのためのテンプレート・システム・コール・ページ・テーブルのコピーをプロセス状態表現で作成し、プロセスごとのエントリ・ページ・テーブルに元のテンプレートと共にユーザ空間マッピングを入力することと、ページ・フォールトが許可されると判定されたときに、カーネル空間メモリへのアクセスを試行したシステム・コール・ハンドラの実行を許可するカーネル空間アドレス範囲のマッピングを追加することとによって作成され得る。方法は、ページ・フォールトが許可されていないと判定されたときに、カーネル空間メモリへのアクセスを試行したシステム・コールを行ったユーザ・プロセスの実行を終了することをさらに含むことができる。方法は、システム・コール・ハンドラの実行が終了したときに、システム・コール・ハンドラの実行中に追加されたカーネル空間アドレス範囲のマッピングを除去することをさらに含むことができる。
【００１０】
実施形態において、システムは、プロセッサと、プロセッサによってアクセス可能なメモリと、メモリに格納されたコンピュータ・プログラム命令とを含むことができ、コンピ

ュータ・プログラム命令は、ユーザ・プロセスがシステム・コールを行うときに、カーネル・モードに切り替え、ユーザ・プロセスのためのシステム・コール・ページ・テーブルを使用してシステム・コール・ハンドラを実行することと、システム・コール・ハンドラが、マップされていないカーネル空間メモリへのアクセスを試行するときに、ページ・フォールトを発生させることと、マップされていないカーネル空間メモリに対して試行されたアクセスが許可されているかどうかを判定することによって、ページ・フォールトを処理することとを実行するように、プロセッサによって実行可能である。

【００１１】

実施形態において、コンピュータ・プログラム製品は、プログラム命令が具現化された非一過性のコンピュータ可読記憶媒体を含むことができ、プログラム命令は、ユーザ・プロセスがシステム・コールを行うときに、カーネル・モードに切り替え、ユーザ・プロセスのためのシステム・コール・ページ・テーブルを使用してシステム・コール・ハンドラを実行することと、システム・コール・ハンドラが、マップされていないカーネル空間メモリへのアクセスを試行するときに、ページ・フォールトを発生させることと、マップされていないカーネル空間メモリに対して試行されたアクセスが許可されているかどうかを判定することによって、ページ・フォールトを処理することとを含み得る方法をコンピュータに実行させるように、コンピュータによって実行可能である。

【００１２】

本発明の詳細は、その構造および動作の両方に関して、添付図面を参照することによって最もよく理解することができる。図中、同一の参照符号および名称は同一の要素を指す。

【図面の簡単な説明】

【００１３】

【図１】本システムおよび方法の実施形態によるシステム・メモリ空間の例示図である。

【図２】本システムおよび方法の実施形態の動作のプロセスの例示的なフロー図である。

【図３】本明細書に記載の実施形態に関与するプロセスが実装され得る、コンピュータ・システムの例示的なブロック図である。

【発明を実施するための形態】

【００１４】

本システムおよび方法の実施形態は、専用アドレス空間でシステム・コールを実行することにより、オペレーティング・システム・カーネル自体の内部に追加のセキュリティ機構を提供して、悪意のあるアプリケーションに可視であるためこのアプリケーションによって悪用可能な共有リソースの量を削減することができる。単一のアドレス空間のみでコードを実行するＬｉｎｕｘカーネルなどの従来のモノリシック・オペレーティング・システム・カーネルは、セキュリティ強化のために複数のアドレス空間でコードを実行するように変更され得る。セキュリティを強化するために、追加の特性を使用することができる。例えば、システム・コール環境において、複数のアドレス空間のコードをカーネル（特権）モードで実行することができ、これを使用して、システム・コールを実行するエンティティがその許可されたアドレス空間の外部へのアクセスを試行した場合に、試行されたアクセスを検出して安全に処理できるようにすることを保証することができる。アドレス空間は、ユーザ・プロセスごとに作成することができ、ユーザ・プロセスに対するセキュリティ特性を強化するために使用することができる。

【００１５】

実施形態において、ユーザ空間アプリケーションがシステム・コールを実行するときに、カーネル・エントリ・コードは、その特定のシステム・コールの実行に適したアドレス空間を選択することができる。システム・コールの実行が、そのアドレス空間の外部のメモリへのアクセスを試行すると、ページ・フォールトが発生する。ページ・フォールト・ハンドラは、要求されたメモリ範囲が、システム・コール・ハンドラ・ルーチンによってアクセス許可されているかどうかを検証し、アクセス許可されていれば、この範囲を専用アドレス空間に追加することができる。システム・コールの完了時に、アドレス空間定義を元の状態に戻すことができる。

【００１６】
システム・メモリ空間１００の例示図を図１に示す。この例は、カーネル・ページ・テーブル１０２、システム・コール・ページ・テーブル１０４、およびユーザ・ページ・テーブル１０６を示す。さらに、後述するテンプレート・システム・コール・ページ・テーブル１２６が存在し得る。この例では、カーネル・ページ・テーブル１０２は、特定のユーザ・プロセスに関連付けることができ、ユーザ空間エントリ１０８、カーネル・エントリ・ページ・テーブル１１０、およびカーネル空間エントリ１１２を含むことができる。ユーザ空間エントリ１０８は、システムがカーネル（特権）・モードで実行されているときに使用するユーザ空間アドレス・マッピングを含むことができる。カーネル空間エントリ１１２は、システムがカーネル（特権）・モードで実行されているときに使用するカーネル空間アドレス・マッピングを含むことができる。カーネル・エントリ・ページ・テーブル１１０は、非特権モードからカーネル（特権）・モードへの切替えを実行するために必要な構造をマップするＣＰＵエントリ領域を含むことができる。ユーザ・ページ・テーブル１０６は特定のユーザ空間プロセスに関連付けることができ、ユーザ空間エントリ１１４は、ユーザ空間プロセスによって使用されるユーザ空間マッピングのみを含むことができる。カーネル・エントリ・ページ・テーブル１１６は、非特権モードからカーネル（特権）・モードへの切替えを実行するために必要な構造をマップするＣＰＵエントリ領域を含むことができる。カーネル・エントリ・ページ・テーブル１１０、１１６、１２０を含むページ・テーブルをこのように分離させると、セキュリティの脆弱性を軽減することができる。

【００１７】
ユーザ・プロセスは、通常、非特権モードで実行され、ユーザ空間エントリ１１４およびカーネル・エントリ・ページ・テーブル１１６のみを含むユーザ・ページ・テーブル１０６にアクセスできる。従来であれば、ユーザ・プロセスがシステム・コールを行うときに、システムは非特権モードからカーネル（特権）・モードに切り替え、プロセスは、ユーザ空間エントリ１０８、カーネル・エントリ・ページ・テーブル１１０、およびカーネル空間エントリ１１２を含むカーネル・ページ・テーブル全体にアクセスできるだろう。しかしながら、本システムおよび方法の実施形態においては、ユーザ・プロセスがシステム・コールを行うときに、システムは非特権モードからカーネル（特権）・モードに切り替えることができ、プロセスはシステム・コール・ページ・テーブル１０４にアクセスできる。システム・コール・ページ・テーブル１０４は、追加のシステム・コール（ｓｙｓｃａｌｌ）エントリ・ページ・テーブル１２２を用いてユーザ・プロセスの可視マッピングを拡大することができる。Ｓｙｓｃａｌｌエントリ・ページ・テーブル１２２は、ユーザ・プロセスが、制限されたカーネル空間アドレス範囲にアクセスして、非特権モードからカーネル（特権）・モードへの切替え後に、コードの制限された追加部分またはデータへのアクセスを実行することを許可することができる。Ｓｙｓｃａｌｌエントリ・ページ・テーブル１２２は、カーネル・コードおよびデータへの悪意のあるアクセスを防止するのに十分に制限されたコードの部分へのアクセスを提供し、しかもユーザ空間エントリ１１８からアクセス可能なコードよりも多くのコードへのアクセスを提供することができる。ｓｙｓｃａｌｌエントリ・ページ・テーブル１２２を通じてコードにアクセスするとき、セキュリティ・コード・ブロック１２４は、行われるいずれのアクセスも安全であることを検証することができる。実施形態において、様々な検証のうちのいずれかをセキュリティ・コード・ブロック１２４によって行うことができる。例えば、実施形態において、セキュリティ・コード・ブロック１２４は、既知の記号のみに対してアクセスが行われることを検証することができる。

【００１８】
本システムおよび方法の実施形態の動作２００のプロセスの例示的なフロー図を図２に示す。図１と組み合わせると最適に見られる。プロセス２００は２０２から始まり、Ｌｉｎｕｘカーネルの初期化中に、図３に示す追加のテンプレート・システム・コール・ページ・テーブル１２６を作成することができる。テンプレート・システム・コール・ページ

・テーブル１２６を使用して、分離されたアドレス空間におけるシステム・コール・ハンドラの実行のために機能的なシステム・コール・ページ・テーブル１０４を作成することができる。テンプレート・システム・コール・ページ・テーブル１２６は、システム・コールの実行時にカーネル（特権）・モードへのエントリを実行するために必要なカーネル・コードおよびデータ、システム・コール・ハンドラへのポインタを有するテーブル、ならびにｓｙｓｃａｌｌエントリ・ページ・テーブル１２２の各システム・コール・ハンドラのエントリ・コードのマッピング１２８を有することができる。さらに、テンプレート・システム・コール・ページ・テーブル１２６は、入力されたカーネル・エントリ・ページ・テーブル１２０と空のユーザ空間ページ・テーブル１１８とを有することができる。

【００１９】

２０４において、プロセスが作成されると、テンプレート・システム・コール・ページ・テーブル１２６をプロセス状態表現にコピーして、オペレーション・システム・コール・ページ・テーブル１０４を形成することができる。入力されたカーネル・エントリ・ページ・テーブル１２０およびｓｙｓｃａｌｌエントリ・ページ・テーブル１２２を含む元のテンプレートに加えて、プロセスごとのシステム・コール・ページ・テーブル１０４の空のユーザ空間ページ・テーブル１１８に、通常のプロセス・ページ・テーブルと同様の方法でユーザ空間マッピングを入力することができる。

【００２０】

２０６において、ユーザ空間プロセスはシステム・コールを実行することができる。ＣＰＵはカーネル（特権）・モードに移行することができ、汎用システム・コール・エントリ・ルーチンを呼び出すことができる。このルーチンは、ＣＰＵページ・テーブル・ポインタ・レジスタを切り替えて、そのプロセスについてシステム・コール・ページ・テーブル１０４を使用することができ、特定のシステム・コール・ハンドラにジャンプすることができる。

【００２１】

２０８において、システム・コール・ハンドラの実行中に、カーネル空間カーネル・コードまたはデータあるいはその両方へのアクセスが、ページ・フォールトを引き起こすことがある。セキュリティ・ブロック１２４のページ・フォールト・ハンドラは、要求されたメモリ・アクセスが安全であるかまたは許可されているかどうかを検証することができる。アクセスが安全であるまたは許可されているとわかった場合、要求されたメモリ範囲のマッピングをプロセスごとのシステム・コール・ページ・テーブル１０４に追加することができる。アクセスが安全でないと考えられるまたは許可されていない場合、ユーザ空間プロセスを終了させることができ、または他のセキュリティ動作を行うことができ、あるいはその両方を行うことができる。

【００２２】

２１０において、システム・コール・ハンドラの実行が終了すると、制御を汎用システム・コール・エントリ・ルーチンに戻すことができる。このルーチンは、プロセスごとのシステム・コール・ページ・テーブル１０４をクリーン・アップし、システム・コール・ハンドラの実行中に追加されたマッピングを除去することができる。

【００２３】

本明細書に記載の実施形態に関与するプロセスが実装され得る、コンピュータ・システム３００の例示的なブロック図を図３に示す。コンピュータ・システム３００は、埋込みプロセッサ、システム・オン・チップ、パーソナル・コンピュータ、ワークステーション、サーバ・システム、およびミニコンピュータまたはメインフレーム・コンピュータなどの１つまたは複数のプログラムされた汎用コンピュータ・システムを使用して、または分散ネットワーク・コンピューティング環境において実装され得る。コンピュータ・システム３００は、１つまたは複数のプロセッサ（ＣＰＵ）３０２Ａ～３０２Ｎ、入出力回路３０４、ネットワーク・アダプタ３０６、およびメモリ３０８を含むことができる。ＣＰＵ３０２Ａ～３０２Ｎは、現在の通信システムおよび方法の機能を実行するためにプログラム命令を実行する。通常、ＣＰＵ３０２Ａ～３０２Ｎは、ＩＮＴＥＬ　ＣＯＲＥ（Ｒ）プ

ロセッサなどの１つまたは複数のマイクロプロセッサである。図３は、コンピュータ・システム３００が単一のマルチプロセッサ・コンピュータ・システムとして実装され、複数のプロセッサ３０２Ａ～３０２Ｎが、メモリ３０８、入出力回路３０４、およびネットワーク・アダプタ３０６などのシステム・リソースを共有する実施形態を示す。しかしながら、本通信システムおよび方法はまた、コンピュータ・システム３００が複数のネットワーク化コンピュータ・システムとして実装される実施形態を含み、これは、シングルプロセッサ・コンピュータ・システム、マルチプロセッサ・コンピュータ・システム、またはこれらの組合せであってよい。

【００２４】

入出力回路３０４は、コンピュータ・システム３００にデータを入力する、またはコンピュータ・システム３００からデータを出力する機能を提供する。例えば、入出力回路として、キーボード、マウス、タッチパッド、トラックボール、スキャナ、アナログ・デジタル・コンバータなどの入力デバイス、ビデオ・アダプタ、モニタ、プリンタなどの出力デバイス、およびモデムなどの入出力デバイスが挙げられる。ネットワーク・アダプタ３０６は、デバイス３００をネットワーク３１０とインタフェースする。ネットワーク３１０は、インターネットを含むがこれに限定されない任意の公衆または専用ＬＡＮまたはＷＡＮであってよい。

【００２５】

メモリ３０８は、コンピュータ・システム３００の機能を実行するためにＣＰＵ３０２によって実行されるプログラム命令、およびＣＰＵ３０２によって使用され処理されるデータを格納する。メモリ３０８として、例えば、ランダム・アクセス・メモリ（ＲＡＭ）、読取り専用メモリ（ＲＯＭ）、プログラマブル読取り専用メモリ（ＰＲＯＭ）、電気的に消去可能なプログラマブル読取り専用メモリ（ＥＥＰＲＯＭ）、フラッシュ・メモリなどの電子メモリ・デバイス、および磁気ディスク・ドライブ、テープ・ドライブ、光ディスク・ドライブなどの電気機械メモリが挙げられ、これらは、インテグレーテッド・ドライブ・エレクトロニクス（ＩＤＥ）・インタフェース、または拡張ＩＤＥ（ＥＩＤＥ）もしくはウルトラダイレクト・メモリ・アクセス（ＵＤＭＡ）などのその変形もしくは拡張、またはスモール・コンピュータ・システム・インタフェース（ＳＣＳＩ）・ベースのインタフェース、またはファスト－ＳＣＳＩ、ワイド－ＳＣＳＩ、ファスト・アンド・ワイド－ＳＣＳＩなどのその変形もしくは拡張、またはシリアル・アドバンスト・テクノロジ・アタッチメント（ＳＡＴＡ）、またはその変形もしくは拡張、またはファイバ・チャネル・アービトレーテット・ループ（ＦＣ－ＡＬ）・インタフェースを使用することができる。

【００２６】

メモリ３０８の内容は、コンピュータ・システム３００が実行するようにプログラムされている機能に応じて変化し得る。図３に示す例において、前述したプロセスの実施形態についてのルーチンおよびデータを表す例示的なメモリ内容が示されている。しかしながら、これらのルーチンは、それらのルーチンに関連したメモリ内容と共に、１つのシステムまたはデバイスに含まれていなくてもよく、周知の工学的考慮事項に基づいて複数のシステムまたはデバイス間に分散させることができることを、当業者は認識するだろう。本通信システムおよび方法は、このような配置のすべてを含むことができる。

【００２７】

図３に示す例において、メモリ３０８は、カーネル空間３１２、ユーザ空間３１４、セキュリティ・ブロック・ルーチン３２４、および他のオペレーティング・システム・ルーチン３２２を含むことができる。カーネル空間３１２は、昇格されたシステム状態に存在するコードおよびデータを含むことができ、保護されたメモリ空間およびハードウェアへのフルアクセスを含むことができる。カーネル空間３１２は、カーネル・ページ・テーブル３１６、システム・コール（ｓｙｓｃａｌｌ）ページ・テーブル３１８、およびテンプレート・システム・コール・ページ・テーブル３２０を含むことができる。カーネル・ページ・テーブル３１６は、図１に示し前述したように、特定のユーザ・プロセスに関連付

けることができ、ユーザ空間ページ・テーブル１０８、カーネル・エントリ・ページ・テーブル１１０、およびカーネル空間ページ・テーブル１１２を含むことができる。システム・コール・ページ・テーブル３１８は、図１に示し前述したように、追加のｓｙｓｃａｌｌエントリ・ページ・テーブル１２２を用いてユーザ・プロセスの可視マッピングを拡大することができる。テンプレート・システム・コール・ページ・テーブル３２０を使用して、前述したように、分離されたアドレス空間におけるシステム・コール・ハンドラの実行のために機能的なシステム・コール・ページ・テーブル１０４を作成することができる。ユーザ空間３１４は、ハードウェアおよびソフトウェアの利用可能なリソースのサブセットのみにアクセス可能な低い権限で実行され得る、ユーザ・アプリケーション、プログラム、タスク、プロセスなどを含むことができる。ユーザ空間３１４は、ユーザ空間の非特権アクセスにマップし得るユーザ・ページ・テーブル３２２を含むことができる。セキュリティ・ブロック・ルーチン３２４は、システム・コール・ページ・テーブル３１８を使用して行われたアクセスが安全であるまたは許可されていることを検証することができる。他のオペレーティング・システム・ルーチン３２２は、追加のシステム機能を提供することができる。

【００２８】

図３に示すように、本通信システムおよび方法は、マルチプロセッサ、マルチタスク、マルチプロセス、またはマルチスレッド・コンピューティング、あるいはその組合せを提供する１つまたは複数のシステム上の実装と、シングル・プロセッサ、シングル・スレッド・コンピューティングのみを提供するシステム上の実装とを含むことができる。マルチプロセッサ・コンピューティングは、複数のプロセッサを使用してコンピューティングを実行することを伴う。マルチタスク・コンピューティングは、複数のオペレーティング・システム・タスクを使用してコンピューティングを実行することを伴う。タスクとは、オペレーティング・システムの概念であり、実行中のプログラムと、オペレーティング・システムによって使用されるブックキーピング情報との組合せを指す。プログラムが実行されるたびに、オペレーティング・システムはそのための新しいタスクを作成する。タスクは、プログラムのための封筒のようなものであり、プログラムをタスク番号で識別し、他のブックキーピング情報をプログラムに付加する。Ｌｉｎｕｘ（Ｒ）、ＵＮＩＸ（Ｒ）、ＯＳ／２（Ｒ）、およびＷｉｎｄｏｗｓ（Ｒ）を含む多くのオペレーティング・システムは、多くのタスクを同時に実行することができ、マルチタスク・オペレーティング・システムと呼ばれる。マルチタスクは、オペレーティング・システムが複数の実行可能ファイルを同時に実行する能力である。各実行可能ファイルは、それ自体のアドレス空間で実行され、すなわち、実行可能ファイルがそれらのメモリのいずれをも共有することはできないことを意味する。このことは、いかなるプログラムも、システム上で実行中の他のプログラムのうちのいずれかの実行に損害を与えることが不可能であるため、有利である。しかしながら、プログラムは、オペレーティング・システムを介して（またはファイル・システムに格納されているファイルを読み取ることによる）以外にいかなる情報も交換することができない。タスクとプロセスという用語はしばしば互換的に使用されるため、マルチプロセス・コンピューティングは、マルチタスク・コンピューティングと同様であるが、一部のオペレーティング・システムではこれら２つを区別する。

【００２９】

本発明は、任意の可能な統合の技術的詳細レベルにおけるシステム、方法、またはコンピュータ・プログラム製品、あるいはその組合せであってよい。コンピュータ・プログラム製品は、プロセッサに本発明の態様を実行させるためのコンピュータ可読プログラム命令を有する１つ（または複数）のコンピュータ可読記憶媒体を含むことができる。コンピュータ可読記憶媒体は、命令実行デバイスによって使用されるように命令を保持し格納することができる有形のデバイスであってよい。

【００３０】

コンピュータ可読記憶媒体は、例えば、電子ストレージ・デバイス、磁気ストレージ・デバイス、光学ストレージ・デバイス、電磁ストレージ・デバイス、半導体ストレージ・

デバイス、またはこれらの任意の適切な組合せであってよいが、これらに限定されない。コンピュータ可読記憶媒体のより具体的な例の非網羅的なリストには、以下のもの、すなわち、携帯型コンピュータ・ディスケット、ハード・ディスク、ランダム・アクセス・メモリ（ＲＡＭ）、読取り専用メモリ（ＲＯＭ）、消去可能なプログラマブル読取り専用メモリ（ＥＰＲＯＭもしくはフラッシュ・メモリ）、スタティック・ランダム・アクセス・メモリ（ＳＲＡＭ）、携帯型コンパクト・ディスク読取り専用メモリ（ＣＤ－ＲＯＭ）、デジタル多用途ディスク（ＤＶＤ）、メモリ・スティック、フロッピ（Ｒ）・ディスク、パンチカードもしくは命令が記録されている溝内の隆起構造などの機械的に符号化されたデバイス、およびこれらの任意の適切な組合せが含まれる。本明細書で使用されるコンピュータ可読記憶媒体は、電波もしくは他の自由に伝搬する電磁波、導波路もしくは他の伝送媒体を伝搬する電磁波（例えば光ファイバ・ケーブルを通過する光パルス）、または電線を介して伝送される電気信号などの、一過性の信号自体であると解釈されるべきではない。

【００３１】

本明細書に記載のコンピュータ可読プログラム命令は、コンピュータ可読記憶媒体からそれぞれのコンピューティング／処理デバイスにダウンロードすることができ、または、ネットワーク、例えばインターネット、ローカル・エリア・ネットワーク、ワイド・エリア・ネットワーク、または無線ネットワーク、あるいはその組合せを介して外部コンピュータもしくは外部ストレージ・デバイスにダウンロードすることができる。ネットワークは、銅伝送ケーブル、光伝送ファイバ、無線伝送、ルータ、ファイアウォール、スイッチ、ゲートウェイ・コンピュータ、またはエッジ・サーバ、あるいはその組合せを含むことができる。各コンピューティング／処理デバイスにおけるネットワーク・アダプタ・カードまたはネットワーク・インタフェースが、ネットワークからコンピュータ可読プログラム命令を受信し、それらのコンピュータ可読プログラム命令を、それぞれのコンピューティング／処理デバイス内のコンピュータ可読記憶媒体に格納するために転送する。

【００３２】

本発明の動作を実行するためのコンピュータ可読プログラム命令は、アセンブラ命令、命令セット・アーキテクチャ（ＩＳＡ）命令、マシン命令、マシン依存命令、マイクロコード、ファームウェア命令、状態設定データ、集積回路の構成データ、あるいはＳｍａｌｌｔａｌｋ（Ｒ）、Ｃ＋＋などのオブジェクト指向プログラミング言語および「Ｃ」プログラミング言語もしくは同様のプログラミング言語などの手続き型プログラミング言語を含む、１つまたは複数のプログラミング言語の任意の組合せで書かれたソース・コードまたはオブジェクト・コードであってよい。コンピュータ可読プログラム命令は、全体的にユーザのコンピュータ上で、一部がユーザのコンピュータ上で、独立型ソフトウェア・パッケージとして、一部がユーザのコンピュータ上かつ一部がリモート・コンピュータ上で、または全体的にリモート・コンピュータもしくはサーバ上で実行することができる。後者のシナリオにおいて、リモート・コンピュータは、ローカル・エリア・ネットワーク（ＬＡＮ）もしくはワイド・エリア・ネットワーク（ＷＡＮ）を含む任意の種類のネットワークを介してユーザのコンピュータに接続することができ、または（例えば、インターネット・サービス・プロバイダを使用してインターネットを介して）外部コンピュータに接続することができる。一部の実施形態において、本発明の態様を実行するために、例えば、プログラマブル論理回路、フィールド・プログラマブル・ゲート・アレイ（ＦＰＧＡ）、またはプログラマブル・ロジック・アレイ（ＰＬＡ）を含む電子回路が、コンピュータ可読プログラム命令の状態情報を利用して電子回路をパーソナライズすることにより、コンピュータ可読プログラム命令を実行することができる。

【００３３】

本発明の実施形態による方法、装置（システム）、およびコンピュータ・プログラム製品のフローチャート図またはブロック図あるいはその両方を参照しながら、本発明の態様について本明細書で説明する。フローチャート図またはブロック図あるいはその両方の各ブロック、およびフローチャート図またはブロック図あるいはその両方におけるブロック

の組合せは、コンピュータ可読プログラム命令によって実装できることが理解されよう。

【００３４】

これらのコンピュータ可読プログラム命令は、コンピュータまたは他のプログラマブル・データ処理装置のプロセッサを介して実行される命令が、フローチャートまたはブロック図あるいはその両方の１つまたは複数のブロックに指定される機能／動作を実施する手段を作り出すべく、汎用コンピュータ、専用コンピュータ、または他のプログラマブル・データ処理装置のプロセッサに提供されてマシンを作り出すものであってよい。これらのコンピュータ可読プログラム命令は、命令が格納されたコンピュータ可読記憶媒体が、フローチャートまたはブロック図あるいはその両方の１つまたは複数のブロックに指定される機能／動作の態様を実施する命令を含んだ製品を含むべく、コンピュータ可読記憶媒体に格納されて、コンピュータ、プログラマブル・データ処理装置、または他のデバイス、あるいはその組合せに特定の方式で機能するように指示できるものであってもよい。

【００３５】

コンピュータ可読プログラム命令は、コンピュータ、他のプログラマブル装置、または他のデバイスで実行される命令が、フローチャートまたはブロック図あるいはその両方の１つまたは複数のブロックに指定される機能／動作を実施するように、コンピュータによって実行されるプロセスを作り出すべく、コンピュータ、他のプログラマブル・データ処理装置、または他のデバイスにロードされ、コンピュータ、他のプログラマブル装置、または他のデバイス上で一連の動作ステップを実行させるものであってもよい。

【００３６】

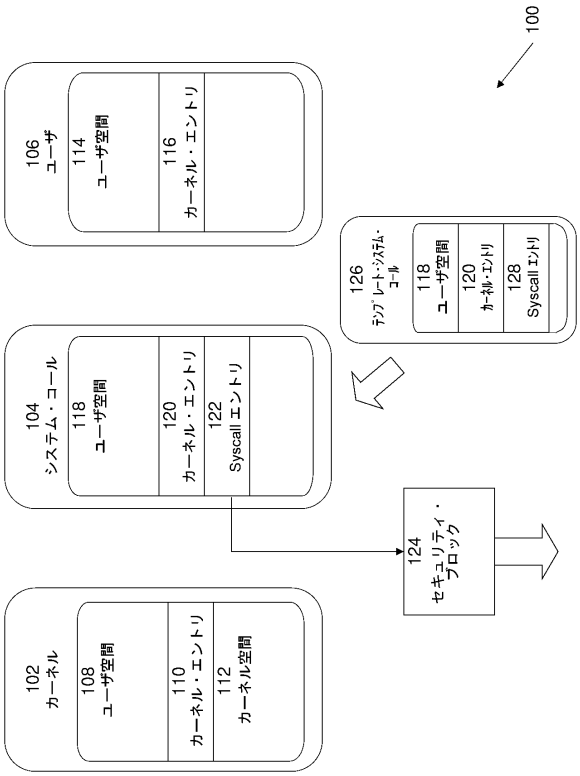
図中のフローチャートおよびブロック図は、本発明の様々な実施形態によるシステム、方法、およびコンピュータ・プログラム製品の可能な実装形態のアーキテクチャ、機能、および動作を示す。これに関して、フローチャートまたはブロック図の各ブロックは、指定された論理機能を実施するための１つまたは複数の実行可能命令を含む、命令のモジュール、セグメント、または部分を表すことができる。一部の代替実装形態において、ブロックに示す機能を、図示する順序以外で行うことができる。例えば、連続して示す２つのブロックを、実際には略同時に実行することができ、または、関与する機能に応じて、それらのブロックを時として逆の順序で実行することができる。また、ブロック図またはフローチャート図あるいはその両方の各ブロック、およびブロック図またはフローチャート図あるいはその両方におけるブロックの組合せは、指定された機能もしくは動作を実行する、または専用ハードウェア命令とコンピュータ命令との組合せを実行する専用ハードウェア・ベースのシステムによって実装することができることにも留意されたい。

【００３７】

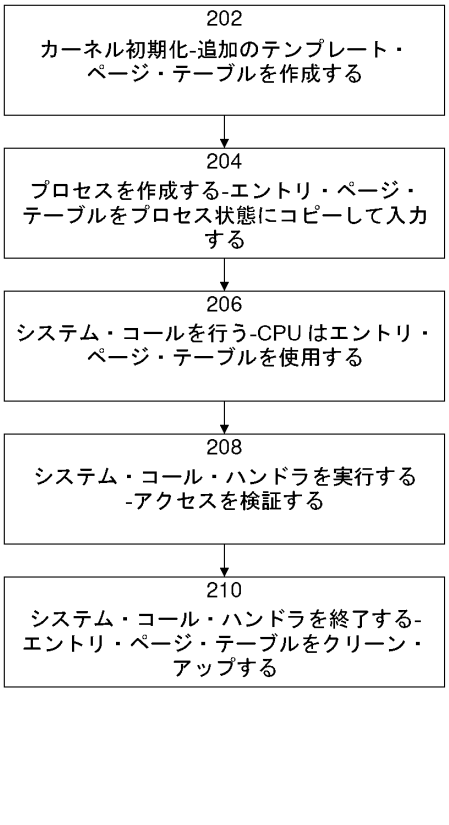
本発明の特定の実施形態について説明したが、説明した実施形態と均等な他の実施形態が存在することが当業者には理解されるだろう。したがって、本発明は特定の例示した実施形態によって限定されるものではなく、添付の特許請求の範囲によってのみ限定されるものであることを理解されたい。

【図面】

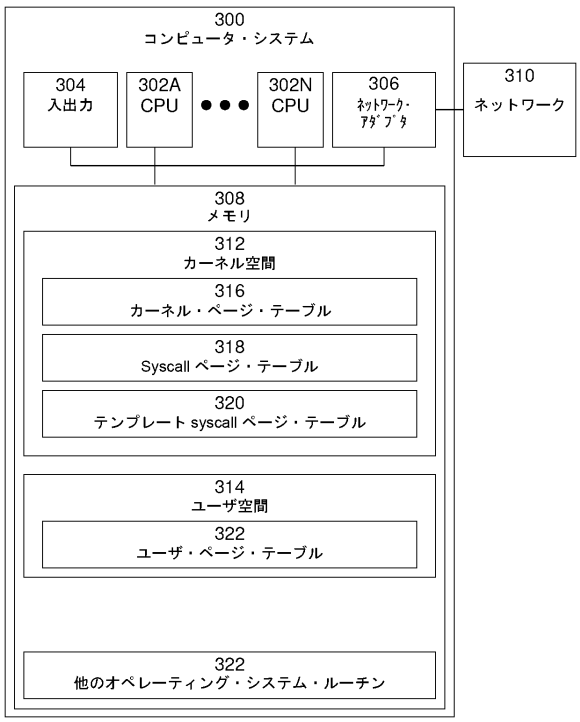
【図１】


100
106
ユーザ
114
ユーザ空間
116
カーネル・エントリ
126
テンプレート・システム・コール
118
ユーザ空間
120
カーネル・エントリ
128
Syscall エントリ
104
システム・コール
118
ユーザ空間
120
カーネル・エントリ
122
Syscall エントリ
124
セキュリティ・ブロック
102
カーネル
108
ユーザ空間
110
カーネル・エントリ
112
カーネル空間

【図２】


202
カーネル初期化-追加のテンプレート・ページ・テーブルを作成する
204
プロセスを作成する-エントリ・ページ・テーブルをプロセス状態にコピーして入力する
206
システム・コールを行う-CPU はエントリ・ページ・テーブルを使用する
208
システム・コール・ハンドラを実行する-アクセスを検証する
210
システム・コール・ハンドラを終了する-エントリ・ページ・テーブルをクリーン・アップする
200

【図３】


300
コンピュータ・システム
304
入出力
302A
CPU
302N
CPU
306
ネットワーク・アダプタ
310
ネットワーク
308
メモリ
312
カーネル空間
316
カーネル・ページ・テーブル
318
Syscall ページ・テーブル
320
テンプレート syscall ページ・テーブル
314
ユーザ空間
322
ユーザ・ページ・テーブル
322
他のオペレーティング・システム・ルーチン

フロントページの続き

(72)発明者　ラポポート、マイケル
イスラエル ハイファ３１９０５ マウント・カーメル ハイファ・ユニバーシティ・キャンパス アイ・ビー・エム イスラエル
(72)発明者　ニーダー、ジョエル、ケリー
イスラエル ハイファ３１９０５ マウント・カーメル ハイファ・ユニバーシティ・キャンパス アイ・ビー・エム イスラエル
(72)発明者　ボトムリー、ジェームス
アメリカ合衆国９８１０１ ワシントン州シアトル スイート１７００ オリーブウェイ１１００
審査官　青木 重徳

(56)参考文献
米国特許出願公開第２０１３／００２４６４６（ＵＳ，Ａ１）
中国特許出願公開第１０１３１５６０８（ＣＮ，Ａ）
小田 逸郎，Ｌｉｎｕｘカーネル２．６解読室 第１０回 プロセス空間の管理，ＵＮＩＸ ＵＳＥＲ，日本，ソフトバンクパブリッシング株式会社，2005年03月01日，第14巻，第3号，pp.117-132
末安 泰三，マンスリーレポート ［Android Watch］ SELinuxによる保護の有効化などAndroid 4.4 のセキュリティ強化点，日経コミュニケーション，日本，日経ＢＰ社，2013年12月01日，第５９９号，pp. 48-49
Daniel Gruss et al.，KASLR is Dead: Long Live KASLR，LNCS, ESSoS 2017: Engineering Secure Software and Systems，2017年06月24日，Vol. 10379，pp. 161-176

(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　２１／５５
Ｇ０６Ｆ　１２／１４
ＪＳＴＰｌｕｓ／ＪＭＥＤＰｌｕｓ／ＪＳＴ７５８０（ＪＤｒｅａｍＩＩＩ）
ＩＥＥＥ Ｘｐｌｏｒｅ
ＴＨＥ ＡＣＭ ＤＩＧＩＴＡＬ ＬＩＢＲＡＲＹ