(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6435834号
(P6435834)
(45)発行日　平成30年12月12日(2018.12.12)　(24)登録日　平成30年11月22日(2018.11.22)

(51)Int.Cl.　　F I
G06F 21/54 (2013.01)　　G06F 21/54
G06F 9/32 (2006.01)　　G06F 9/32 383Z

請求項の数 6　(全 21 頁)

(21)出願番号　特願2014-251644(P2014-251644)
(22)出願日　平成26年12月12日(2014.12.12)
(65)公開番号　特開2016-115033(P2016-115033A)
(43)公開日　平成28年6月23日(2016.6.23)
審査請求日　平成29年11月13日(2017.11.13)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100094525
弁理士　土井　健二
(74)代理人　100094514
弁理士　林　恒徳
(72)発明者　古川　和快
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　兒島　尚
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

最終頁に続く

(54)【発明の名称】命令実行制御装置、命令実行制御方法

(57)【特許請求の範囲】
【請求項１】
コール命令を含む第１の命令群と、リターン命令を含む第２の命令群とを記憶する記憶部と、
判定部を含み命令をデコードする命令デコード部と、
分岐部を含み前記命令を実行する命令実行部と、を有し、
前記分岐部は、前記第１の命令群から前記コール命令によって呼び出された前記リターン命令の前記命令実行部による実行時に、前記第１の命令群への戻り先アドレスの命令の判定を前記判定部に指示し、
前記判定部は、前記命令デコード部によってデコードされた命令であって前記リターン命令による前記戻り先アドレスの命令が識別情報を有するか否かを判定し、前記識別情報を有する場合に、前記第１の命令群の処理を続行し、前記識別情報を有しない場合に、命令実行処理を停止する、
命令実行制御装置。
【請求項２】
請求項１において、
前記判定部は、前記戻り先アドレスの命令に前記識別情報が付加されているか否かに基づいて、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御装置。
【請求項３】

請求項１において、
前記判定部は、前記戻り先アドレスの命令が前記識別情報であるか否かに基づいて、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御装置。
【請求項４】
請求項１乃至３のいずれかにおいて、
前記判定部は、さらに、前記戻り先アドレスの命令が前記識別情報を有する場合に前記識別情報が所定の情報であるか否かを判定し、前記識別情報が前記所定の情報である場合、前記第１の命令群の処理を続行し、前記所定の情報ではない場合、前記命令実行処理を停止する、
命令実行制御装置。
【請求項５】
請求項１乃至４のいずれかにおいて、
前記判定部は、さらに、前記第２の命令群の前記リターン命令が検証用のリターン命令であるか否かを判定し、前記検証用のリターン命令である場合に、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定し、前記検証用のリターン命令ではない場合に、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定せず前記第１の命令群の処理を続行する、
命令実行制御装置。
【請求項６】
命令を実行する命令実行部に含まれる分岐部が、第１の命令群に含まれるコール命令によって呼び出された、前記第２の命令群に含まれるリターン命令の前記命令実行部による実行時に、前記第１の命令群への戻り先アドレスの命令の判定を、前記命令をデコードする命令デコード部に含まれる判定部に指示し、
前記判定部が、前記命令デコード部によってデコードされた命令であって前記リターン命令による前記戻り先アドレスの命令が識別情報を有するか否かを判定し、
前記判定部が、前記識別情報を有する場合に、前記第１の命令群の処理を続行し、前記識別情報を有しない場合に、命令実行処理を停止する、
命令実行制御方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、命令実行制御装置、命令実行制御方法に関する。
【背景技術】
【０００２】
インターネットの発達に伴い、プロセッサへの不正なアクセスが増加している。この不正なアクセスを防止するため、セキュリティ機能を設けている。
【０００３】
プロセッサへの不正なアクセスには、例えば、バッファオーバーフローによって、攻撃コードをメモリ上に展開し、攻撃コードへ実行を遷移させる攻撃手法がある。また、この攻撃へのセキュリティ対策として、データ実行防止（Data Execution Prevention：ＤＥＰ）機能がある。ＤＥＰ機能は、データメモリ領域上でのプログラムの実行を禁止することにより、バッファオーバーフローによる攻撃を困難にする。
【０００４】
一方、ＤＥＰ機能による制限を回避する攻撃手法として考え出された、ＲＯＰ（Return Oriented Programming：ＲＯＰ）攻撃がある。ＲＯＰ攻撃は、スタックオーバーフローなどのプログラムの脆弱性を利用してスタック情報を改竄し、プログラムの実行遷移を制御するプログラミング手法である。
【０００５】
ＲＯＰ攻撃については、例えば、特許文献１、２に記載される。

【先行技術文献】
【特許文献】
【０００６】
【特許文献１】特開２０１３－２２８９５７号公報
【特許文献２】特開２０１１－００８７７８号公報
【発明の概要】
【発明が解決しようとする課題】
【０００７】
ＲＯＰ攻撃は、スタックを改ざんする。そして、ＲＯＰ攻撃は、改ざんしたスタックを実行することによって、実行したい疑似プログラムと同等の処理を実現する。ＲＯＰ攻撃により、攻撃者は、スタックを改ざんすることによって、カーネル空間で、攻撃プログラムを実行することができる。
【０００８】
１つの側面は、本発明は、ＲＯＰ攻撃を検出し、被害を未然に防ぐ命令実行制御装置、命令実行制御方法を提供することを目的とする。
【課題を解決するための手段】
【０００９】
第１の側面は、コール命令を含む第１の命令群と、リターン命令を含む第２の命令群とを記憶する記憶部と、前記第１の命令群から前記コール命令によって呼び出された前記第２の命令群の実行時に、前記第２の命令群の前記リターン命令による、前記第１の命令群への戻り先アドレスの命令が識別情報を有するか否かを判定し、前記識別情報を有する場合に、前記第１の命令群の処理を続行し、前記識別情報を有しない場合に、命令実行処理を停止する処理部、とを有する。
【発明の効果】
【００１０】
第１の側面によれば、ＲＯＰ攻撃を検出し、被害を未然に防ぐ。
【図面の簡単な説明】
【００１１】
【図１】本実施の形態におけるＲＯＰ攻撃を説明する図である。
【図２】図１で説明した、ＲＯＰ攻撃におけるガジェットと疑似プログラムとの対応を説明する図である。
【図３】第１の実施の形態における、戻り先ラベルフィールドを付加した、アセンブリコードの一例を示す図である。
【図４】第１の実施の形態における、戻り先ラベルを付加した、アセンブリコードの一例を示す図である。
【図５】第１の実施の形態における情報処理装置のハードウェア構成を示す図である。
【図６】図５に示した情報処理装置の命令実行制御処理の流れを説明する図である。
【図７】図５に示した情報処理装置の処理を、図１に示したスタック領域が改ざんされていない場合に基づいて説明する図である。
【図８】図５に示した情報処理装置の処理を、図１に示したスタック領域が改ざんされている場合に基づいて説明する図（その１）である。
【図９】図５に示した情報処理装置の処理を、図１に示したスタック領域が改ざんされている場合に基づいて説明する図（その２）である。
【図１０】第２の実施の形態におけるアセンブリコードの一例を示す図である。
【図１１】第２の実施の形態における情報処理装置のハードウェア構成を示す図である。
【発明を実施するための形態】
【００１２】
以下、図面にしたがって本発明の実施の形態について説明する。ただし、本発明の技術的範囲はこれらの実施の形態に限定されず、特許請求の範囲に記載された事項とその均等物まで及ぶものである。

【００１３】
[第１の実施の形態]
（ＲＯＰ攻撃）
図１は、本実施の形態におけるＲＯＰ（Return-Oriented Programming：ＲＯＰ）攻撃を説明する図である。ＲＯＰ攻撃は、スタックオーバーフローなどのプログラムの脆弱性を利用して、スタック情報を改ざんし、実行する命令の遷移を制御するプログラミング手法である。
【００１４】
図１は、ＲＯＰ攻撃によって疑似的に実行される疑似プログラムｐｂと、情報処理装置（プロセッサ、命令実行制御装置）のメモリ領域のカーネル空間内のプログラムｐａと、メモリ領域内のスタック領域３０とを示す。情報処理装置の構成は、図５にしたがって後述する。
【００１５】
図１に示す、情報処理装置のメモリ領域（図示せず）のカーネル空間内のプログラムｐａは、カーネル空間で動作するプログラムを示す。メモリ領域は、カーネル空間と、ユーザ空間（図示せず）とを含む。
【００１６】
カーネル空間は、ＯＳ（Operating System：ＯＳ）のカーネルのプログラムｐａを格納する仮想メモリ領域である。また、カーネルは、ＯＳの基本機能を実装するソフトウェアである。カーネルのプログラムｐａは、例えば、図１に示すライブラリプログラム（以下、ライブラリ）Ａ～Ｃ、及び、ＡＰＩ（Application Programming Interface：ＡＰＩ）を含む。一方、ユーザ空間では、カーネル以外のユーザのアプリケーションが動作する。カーネル空間で動作するプログラムｐａは、一定以上の権限を有する。したがって、ユーザ空間で動作するプログラム（図示せず）は、カーネル空間で動作するプログラムｐａにアクセスできない。
【００１７】
カーネル空間で不正なプログラムが動作した場合、情報処理装置自体に障害が生じてしまうことがある。これに対し、ユーザ空間で不正なプログラムが動作したとしても、ユーザ空間で動作するプログラムの権限が低いことから、情報処理装置に生じる影響は小さい。
【００１８】
図１に示すスタック領域３０は、例えば、プログラムのリターンアドレスをスタックする領域である。リターンアドレスは、メインルーチンからのサブルーチンが呼び出された場合、サブルーチンの処理後に遷移するべきメインルーチンの命令アドレス（以下、戻り先アドレス）を示す。情報処理装置は、サブルーチンのリターン命令（return）の処理時に、スタック領域３０から、メインルーチンへの戻り先アドレスを取得する。
【００１９】
なお、図１に示すスタック領域３０は、改ざんされている。スタック領域３０が改ざんされていない場合、図１に示す、通常スタックを保持する。これに対し、改ざんされているスタック領域３０は、ガジェットｇａ１～ｇａ３の呼出しスタックや、ＡＰＩ呼出しスタックを保持する。ガジェットｇａ１～ｇａ３（ガジェットｇａともいう）は、プログラムｐａのライブラリＡ～Ｃ及びＡＰＩの一部の命令列であって、疑似プログラムｐｂを構成するための命令列である。また、ガジェットｇａ１～ｇａ３は、リターン命令を末尾に有する命令列である。
【００２０】
図１の例によると、プログラムｐａのライブラリＡが含むガジェットｇａ１は、命令「mov G0, 0xffff」と命令「return」とを有する。また、プログラムｐａのライブラリＢが含むガジェットｇａ２は、命令「mov G1, 0xffff」と命令「return」とを有する。また、プログラムｐａのライブラリＣが含むガジェットｇａ３は、命令「mul G0, G1」と命令「return」とを有する。

【００２１】
ＲＯＰの攻撃者は、スタック領域３０が、ガジェットｇａ１～ｇａ３の先頭アドレス（ガジェット呼出しスタック）を有するように改ざんする。例えば、ＲＯＰの攻撃者は、ガジェットｇａ１、ガジェットｇａ２、ガジェットｇａ３の順に、ガジェット呼出しスタックが取得されるように、スタック領域３０を改ざんする。これにより、ＲＯＰの攻撃者は、情報処理装置に、ガジェットｇａ１～３を、連続的に呼び出させ実行させることができる。
【００２２】
具体的に、情報処理装置は、リターン命令を処理する時、改ざんされたスタック領域３０から、ガジェットｇａ１の先頭アドレスを読み出す。そして、情報処理装置は、ガジェットｇａ１の命令「mov G0, 0xffff」、命令「return」を処理する。また、情報処理装置は、ガジェットｇａ１の命令「return」の処理時、ガジェットｇａ２の先頭アドレスをスタック領域３０から読み出し、ガジェットｇａ２の命令「mov G1, 0xffff」、命令「return」を処理する。同様にして、情報処理装置は、ガジェットｇａ２の命令「return」の処理時、ガジェットｇａ３の先頭アドレスをスタック領域３０から読み出し、ガジェットｇａ３の命令「mul G0, G1」、命令「return」を処理する。
【００２３】
これにより、ＲＯＰの攻撃者は、カーネル空間で、命令列「mov G0, 0xffff、mov G1, 0xffff、mul G0, G1…」を、情報処理装置に実行させることができる。つまり、ＲＯＰの攻撃者は、ガジェットｇａ１～ｇａ３とスタック領域３０を利用することにより、図１に示す疑似プログラムｐｂと同等の処理を、情報処理装置に実行させることができる。
【００２４】
図２は、図１で説明した、ＲＯＰ攻撃におけるガジェットｇａ１～ｇａ３と疑似プログラムｐｂとの対応を説明する図である。図２において、図１で示したものと同一のものは、同一の記号で示す。
【００２５】
図１で前述したとおり、ガジェットｇａ１～ｇａ３は、プログラムｐａの末尾にリターン命令を含む命令列である。図２に示すように、単体のガジェットｇａ１～ｇａ３は、多くの命令を含んでいない。しかしながら、ＲＯＰの攻撃者は、複数のガジェットｇａ１～ｇａ３を組み合わせることによって、疑似的に、疑似プログラムｐｂと同等の処理を実現することができる。
【００２６】
また、ＲＯＰの攻撃者は、カーネル空間で動作するプログラムｐａの一部の命令列をガジェットｇａ１～ｇａ３として悪用する。これにより、ＲＯＰの攻撃者は、昇格した権限にしたがって、カーネル空間で任意のプログラムを実行させることができる。したがって、ＲＯＰ攻撃によって、例えば、カーネルに被害を与えることが可能になる。
【００２７】
（本実施の形態の概要）
図１、図２に示したように、ＲＯＰ攻撃では、命令の実行が、通常はサブルーチンから戻り先にならないプログラムｐａ内の途中の命令に遷移する。したがって、本実施の形態における情報処理装置は、命令の実行が、サブルーチンコールからの戻り先アドレスが示す命令とは異なる命令に遷移するＲＯＰ攻撃の特性を利用し、ＲＯＰ攻撃を検出する。即ち、情報処理装置は、リターン命令による遷移先の命令が、戻り先アドレスの命令ではない場合に、ＲＯＰ攻撃を検出する。
【００２８】
具体的に、本実施の形態において、命令列のうち、サブルーチンからの戻り先アドレスの命令は、戻り先のアドレスであることを示す戻り先ラベル（識別情報）を有する。そして、情報処理装置は、第１のルーチン（命令群）のコール命令によって呼び出された第２のルーチン（命令群）の実行時に、第２のルーチンのリターン命令による第１のルーチンへの戻り先アドレスの命令が戻り先ラベルを有するか否かを判定する。情報処理装置は、

戻り先アドレスの命令が戻り先ラベルを有する場合に、第１のルーチンの処理を続行し、戻り先ラベルを有しない場合に命令実行処理を停止する。

【００２９】

戻り先アドレスの命令が戻り先ラベルを有しない場合、不正なアドレスに遷移したことを示し、スタック領域３０が改ざんされていることを示す。したがって、情報処理装置は、命令の実行処理を停止する。このように、情報処理装置は、サブルーチンからの、通常の戻り先アドレスとは異なる、不正なアドレスへの遷移を検出することでＲＯＰ攻撃を検出し、ＲＯＰ攻撃による被害を未然に防ぐ。

【００３０】

前述したとおり、情報処理装置は、戻り先アドレスの命令が戻り先ラベルを有しない場合、命令の実行処理を停止する。これは、ＲＯＰの攻撃者が、戻り先アドレスの命令が戻り先ラベルを有するようにスタック領域３０を改ざんすれば、情報処理装置は、命令の実行処理を停止しないことを意味する。

【００３１】

しかしながら、カーネル空間のプログラムｐａが有する命令列のうち、戻り先ラベルを有する命令を含み、ＲＯＰの攻撃者が利用したい命令を含む命令列は少ない。より具体的に、戻り先ラベルを有する命令を先頭とし、ＲＯＰの攻撃者が利用したい命令を含み、リターン命令を末尾に有する命令列は、少ない。即ち、戻り先ラベルの制約があることにより、ガジェットｇａとして利用可能な命令列は、大幅に限定される。

【００３２】

これにより、限定されたガジェットｇａに基づいて、情報処理装置に実行させたい任意の疑似プログラムｐｂを構成することは、より困難になる。したがって、本実施の形態の情報処理装置は、ＲＯＰ攻撃を検出しＲＯＰ攻撃による被害を未然に防ぐとともに、ガジェットｇａとして使用可能な命令列を制限することによって、ＲＯＰ攻撃の容易性を低くすることができる。

【００３３】

（アセンブリコード）

第１の実施の形態におけるプログラムｐａの一例として、図３、図４にしたがって、アセンブリコードの一例を説明する。図３、図４に示すアセンブリコードは、図１、図２に示したプログラムｐａの一例であって、アセンブリコードで記述されたプログラムである。図３、図４のアセンブリコードは、戻り先ラベルの付加の形態が異なる。

【００３４】

図３は、第１の実施の形態における、戻り先ラベルフィールドを付加した、アセンブリコードｐｘの一例を示す図である。図３に示すアセンブリコードｐｘは、命令１「call lib_b」、命令２「mov G0, 0xffff, ret」、命令３「mov G1, 0xffff」、命令４「mov G2, 0xffff」、命令「…」、命令ｎ「return」を有する。

【００３５】

図３に示すアセンブリコードｐｘをアセンブリした、バイナリエンコードの各命令は、戻り先ラベルのフィールドを有する。戻り先ラベルフィールドは、戻り先ラベルを付加するフィールドである。命令２「mov G0, 0xffff, ret」における３つ目の領域（「ret」の領域）は、戻り先ラベルフィールドを示す。また、命令２の値「ret」は、戻り先ラベルの値を示す。プログラムの作成者は、サブルーチンを呼び出す命令１「call lib_b」の戻り先アドレス（復帰先アドレス）の、命令２の戻り先ラベルフィールドに、戻り先ラベルを示す値「ret」を付加する。

【００３６】

したがって、図３の例では、情報処理装置は、戻り先アドレスの命令に、戻り先ラベルが付加されているか否かに基づいて、戻り先アドレスの命令が戻り先ラベルを有するか否かを判定する。

【００３７】

コンパイラは、図３に示すアセンブリコードｐｘをアセンブリし、情報処理装置で実行

可能なバイナリエンコードを生成する。このとき、コンパイラは、命令２以外の命令１、命令３～命令ｎに、戻り先ラベルフィールドのみを付加する。したがって、バイナリエンコードの命令２以外の命令１、命令３～命令ｎの戻り先ラベルフィールドは、戻り先ラベルの値を有しない。一方、バイナリエンコードの命令２の戻り先ラベルフィールドは、例えば、１ビットの戻り先ラベルの値を有する。

【００３８】

このように、例えば、コンパイラは、各命令に、戻り先ラベルフィールドを付加する。これにより、プログラムの作成者は、戻り先ラベルの付加のために、アセンブリコードｐｘの命令２以外のすべての命令に対して、戻り先ラベルのフィールドを追加する必要がない。したがって、プログラム（アセンブリコードｐｘ）に対する戻り先ラベルの付加が容易になる。

【００３９】

または、コンパイラは、コンパイル時に、サブルーチンを呼び出す命令１「call lib_b」の戻り先アドレスである命令２を検知し、命令２に「戻り先ラベル（ret）」を付加してもよい。このように、コンパイラが、サブルーチンを呼び出す命令の戻り先アドレスが示す命令を検知することにより、プログラムへの戻り先ラベルの付加が、より容易になる。

【００４０】

図４は、第１の実施の形態における、戻り先ラベルを付加した、アセンブリコードｐｙの一例を示す図である。図３のアセンブリコードｐｘに対して、図４のアセンブリコードｐｙは、戻り先アドレスに、命令「LABEL_RET（戻り先ラベル）」を有する。

【００４１】

図４に示すアセンブリコードｐｙは、命令１１「call lib_b」、命令１２「LABEL_RET（戻り先ラベル）」、命令１３「mov G0, 0xffff, ret」、命令１４「mov G1, 0xffff」、命令１５「mov G2, 0xffff」、命令「…」、命令１ｎ「return」を有する。命令１２「LABEL_RET（戻り先ラベル）」は、例えば、１ビットの値であって、実行されない命令である。

【００４２】

プログラムの作成者は、サブルーチンを呼び出す命令１１「call lib_b」の戻り先アドレス（復帰先アドレス）に、命令（ラベル）「LABEL_RET」を追加する。したがって、図４の例では、情報処理装置は、戻り先アドレスの命令が戻り先ラベルであるか否かに基づいて、戻り先アドレスの命令が戻り先ラベルを有するか否かを判定する。

【００４３】

コンパイラは、図４に示すアセンブリコードｐｙをアセンブリし、情報処理装置で実行可能なバイナリエンコードを生成する。なお、コンパイラは、サブルーチンを呼び出す命令１１「call lib_b」の戻り先アドレスを検知し、戻り先ラベルを追加してもよい。コンパイラが、サブルーチンを呼び出す命令の戻り先アドレスを検知し、戻り先ラベルを追加することにより、プログラムに対する戻り先ラベルの付加が、より容易になる。

【００４４】

なお、図４の例において、アセンブリコードｐｙは、戻り先ラベルの代わりに、ＮＯＰ命令等の所定の命令を有してもよい。戻り先ラベルと同様にして、情報処理装置は、所定の命令を実行しない。

【００４５】

なお、図３、図４に示すアセンブリコードｐｘ、ｐｙに対する、戻り先ラベルフィールド、または、戻り先ラベル（所定の命令）の追加による、プログラムコードの増加量は５％程度である。したがって、戻り先ラベルフィールド、または、戻り先ラベル（所定の命令）の追加によるプログラムコードの増加量は、多くはない。

【００４６】

なお、図３、図４では、アセンブリコードのプログラムを例示したが、戻り先ラベルを付加する対象のプログラムは、アセンブリコードのプログラムに限定されるものではない

。
【００４７】
（情報処理装置のハードウェア構成）
図５は、第１の実施の形態における情報処理装置のハードウェア構成を示す図である。図５に示す情報処理装置１００は、例えば、プログラムカウンタ（Program Counter：ＰＣ）１０、命令メモリ１１、命令フェッチ部１２、命令デコード部１３、命令実行部１４等を有する。
【００４８】
プログラムカウンタ１０は、次に実行すべき命令のアドレスを格納するレジスタである。命令メモリ１１は、情報処理装置１００が処理する命令を格納する領域である。命令メモリ１１は、例えば、ＳＲＡＭ（Static Random Access Memory：ＳＲＡＭ）である。命令フェッチ部１２は、プログラムカウンタ１０の値に基づいて、命令を、命令メモリ１１から読み出し、命令デコード部１３に出力する。命令デコード部１３は、命令フェッチ部１２によって取得された命令をデコードし、命令実行部１４に出力する。
【００４９】
命令実行部１４は、演算部（図示せず）と分岐部１９を有する。演算部は、例えば、加減算、乗算といった算術演算や、論理演算などの演算処理を実行し、演算結果をレジスタ（図示せず）等に書き込む。また、分岐部１９は、例えば、リターン命令による分岐時に、スタック領域（図１）３０から戻り先アドレスを取り出し、プログラムカウンタ１０に設定する。なお、図５にはスタック領域３０を図示していないが、スタック領域３０は、例えば、メインメモリ（図示せず）、または、情報処理装置１００のキャッシュメモリ（図示せず）に含まれる。分岐部１９は、メインメモリやキャッシュメモリと接続する。
【００５０】
また、本実施の形態における命令デコード部１３は、戻り先チェック部１７を有する。さらに、本実施の形態における情報処理装置１００は、戻り先ラベル検証部１８を有する。
【００５１】
分岐部１９は、リターン命令による分岐時に戻り先チェック部１７を有効化する。有効化されている場合、戻り先チェック部１７は、命令デコード部１３がデコードした命令が、戻り先ラベルを有するか否かを判定し、自身を無効化する。命令が戻り先ラベルを有する場合、戻り先チェック部１７は、戻り先ラベルを、戻り先ラベル検証部１８に出力する。一方、命令が戻り先ラベルを有しない場合、戻り先チェック部１７は、例外割り込みを発生させる。
【００５２】
戻り先チェック部１７は、例えば、ＡＮＤ回路（図示せず）を有する。ＡＮＤ回路は、例えば、戻り先ラベルの値を示す信号と、戻り先チェック部１７が有効か無効かを示す信号とを入力とする。ＡＮＤ回路は、戻り先チェック部１７が有効であって、かつ、戻り先ラベルの値がある場合に、戻り先ラベルの値を出力する。
【００５３】
戻り先ラベル検証部１８は、戻り先ラベルが適切な値であるか否かを検証する。戻り先ラベルが適切な値ではない場合、戻り先ラベル検証部１８は、例外割り込みを発生させる。戻り先ラベル検証部１８は、例えば、比較回路（図示せず）を有する。比較回路は、戻り先チェック部１７が出力した戻り先ラベルの値と、適切な戻り先ラベルの値とを比較し、比較結果を出力する。
【００５４】
（命令実行制御処理の流れ）
図６は、図５に示した情報処理装置１００の命令実行制御処理の流れを説明する図である。
【００５５】
Ｓ１１：命令実行部１４は、実行対象の命令がリターン命令か否かを判定する。

【００５６】
Ｓ１２：リターン命令を実行する場合（Ｓ１１のＹｅｓ）、命令実行部１４の分岐部１９は、リターン命令の戻り先アドレスをスタック領域３０から取得し、プログラムカウンタ１０に設定する。また、分岐部１９は、戻り先チェック部１７を有効化する。
【００５７】
Ｓ１３：命令フェッチ部１２は、プログラムカウンタ１０が示す命令を、命令メモリ１１から読み出し、命令デコード部１３に出力する。命令デコード部１３は、命令フェッチ部１２によって取得された命令をデコードする。
【００５８】
Ｓ１４：命令デコード部１３は、戻り先チェック部１７が有効化されているか否かを判定する。なお、戻り先チェック部１７が無効である場合（Ｓ１４のＮｏ）、命令デコード部１３は、デコードした命令を、命令実行部１４に出力する。
【００５９】
Ｓ１５：一方、戻り先チェック部１７が有効化されている場合（Ｓ１４のＹｅｓ）、命令デコード部１３は、戻り先チェック部１７を無効化する。
【００６０】
Ｓ１６：工程Ｓ１５の後、戻り先チェック部１７は、工程Ｓ１３でデコードした命令が戻り先ラベルを有するか否かを判定する。
【００６１】
Ｓ１７：命令が戻り先ラベルを有する場合（Ｓ１６のＹｅｓ）、戻り先チェック部１７は、戻り先ラベルを、戻り先ラベル検証部１８に出力する。
【００６２】
Ｓ１８：戻り先ラベル検証部１８は、戻り先チェック部１７が出力した、戻り先ラベルが所定の情報であるか否かを判定する。即ち、戻り先ラベル検証部１８は、戻り先ラベルが適切な値であるか否かを判定する。戻り先ラベルが所定の情報であるか否かを判定することにより、より厳密に、戻り先アドレスが適切な値であるか否かを判定可能になる。
【００６３】
例えば、戻り先ラベル検証部１８は、戻り先チェック部１７が出力した戻り先ラベルと、予め定めた所定の情報とを比較し、一致するか否かに基づいて、戻り先ラベルが適切であるか否かを判定する。戻り先ラベルが適切である場合（Ｓ１８のＹｅｓ）、命令デコード部１３は、デコードした命令を命令実行部１４に出力する。
【００６４】
Ｓ１９：一方、命令が戻り先ラベルを有していない場合（Ｓ１６のＮｏ）、戻り先チェック部１７は、例外割り込みを発生させる。または、戻り先ラベルが不正なラベルである場合（Ｓ１８のＮｏ）、命令デコード部１３は、例外割り込みを発生させる。
【００６５】
（スタック領域が改ざんされていない場合）
図７は、図５に示した情報処理装置１００の処理を、図１に示したスタック領域３０が改ざんされていない場合に基づいて説明する図である。図７において、図１で示したものと同一のものは、同一の記号で示す。図７は、プログラムｐｙ－１と、プログラムｐｙ－１が呼び出すライブラリ「lib_b」ｐｙ－２と、スタック領域３０とを示す。プログラムｐｙ－１、及び、ライブラリ「lib_b」ｐｙ－２は、カーネル空間で動作するプログラムである。
【００６６】
図７に示すプログラムｐｙ－１は、命令３１～３７を有する。アドレス「0xFF01」が示す命令３１は、命令「call lib_b」である。アドレス「0xFF02」が示す命令３２は、命令（ラベル）「LABEL_RET（戻り先ラベル）」である。アドレス「0xFF03」が示す命令３３は、命令「mov G0, 0xffff」である。アドレス「0xFF04」が示す命令３４は、命令「mov G1, 0xffff」である。アドレス「0xFF05」が示す命令３５は、命令「mov G2, 0xffff」である。アドレス「0xFF07」が示す命令３７は、命令「return」である。

【００６７】
つまり、図７に示すプログラムｐｙ－１は、ライブラリ「lib_b」ｐｙ－２を呼び出した後、レジスタＧ０～Ｇ２が保持する値を順次、移動するプログラムである。
【００６８】
命令３１「call lib_b」の呼出し時、命令実行部１４は、スタック領域３０に、戻り先アドレス「0xFF02」を追加する。そして、命令実行部１４は、ライブラリ「lib_b」ｐｙ－２の命令を順次、実行し、アドレス「0xFFF7」が示す、末尾のリターン命令を処理する（図６のＳ１１のＹｅｓ）。このとき、分岐部１９は、スタック領域３０から戻り先アドレス「0xFF02」を取得し、プログラムカウンタ１０に設定するとともに、戻り先チェック部１７を有効化する（Ｓ１２）。
【００６９】
命令フェッチ部１２は、プログラムカウンタ１０が示す、命令（ラベル）「LABEL_RET（戻り先ラベル）」を取得し、命令デコード部１３に出力する。命令デコード部１３は、命令（ラベル）「LABEL_RET」をデコードし（Ｓ１３）、戻り先チェック部１７が有効化されているか否かを判定する（Ｓ１４）。
【００７０】
工程Ｓ１２によって、戻り先チェック部１７は有効化されている（Ｓ１４のＹｅｓ）。したがって、命令デコード部１３は、戻り先チェック部１７を無効化するとともに（Ｓ１５）、命令（ラベル）「LABEL_RET」が戻り先ラベルを有するか否かを判定する（Ｓ１６）。命令（ラベル）「LABEL_RET」は戻り先ラベルであるため（Ｓ１６のＹｅｓ）、戻り先ラベル検証部１８は、戻り先ラベルが適切な値であるか否かを判定する（Ｓ１７、Ｓ１８）。
【００７１】
戻り先ラベル検証部１８は、命令（ラベル）「LABEL_RET」が、適切なラベル「LABEL_RET」と一致するか否かを判定する。図７の例に示す命令（ラベル）「LABEL_RET」は、適切な値であるため（Ｓ１８のＹｅｓ）、命令デコード部１３は、例外割り込みを発生させない。なお、命令実行部１４は、命令（ラベル）「LABEL_RET」を実行しない。
【００７２】
次に、命令フェッチ部１２は、インクリメントしたプログラムカウンタ１０が示す、アドレス「0xFF03」の命令「mov G0, 0xffff」を取得する。命令デコード部１３は、命令「mov G0, 0xffff」をデコードし（Ｓ１３）、戻り先チェック部１７が有効化されているか否かを判定する（Ｓ１４）。前回、処理した命令がリターン命令ではないことから、戻り先チェック部１７は有効化されていない（Ｓ１４のＮｏ）。したがって、命令デコード部１３は、デコードした命令を、命令実行部１４に出力する。
【００７３】
（スタック領域が改ざんされている場合）
図８は、図５に示した情報処理装置１００の処理を、図１に示したスタック領域３０が改ざんされている場合に基づいて説明する図（その１）である。図８において、図１で示したものと同一のものは、同一の記号で示す。図８は、ライブラリ「lib_a」ｐｙ－１１と、ライブラリ「lib_b」ｐｙ－１２と、スタック領域３０とを示す。ライブラリ「lib_a」ｐｙ－１１、及び、ライブラリ「lib_b」ｐｙ－１２は、カーネル空間で動作するプログラムである。
【００７４】
図８に示すライブラリ「lib_a」ｐｙ－１１は、命令４１～４６を有する。アドレス「0xFF01」が示す命令４１は、命令「mov G3, 0xffff」である。アドレス「0xFF02」が示す命令４２は、命令「mov G0, 0xffff」である。アドレス「0xFF03」が示す命令４３は、命令「mov G1, 0xffff」である。アドレス「0xFF04」が示す命令４４は、命令「mov G2, 0xffff」である。アドレス「0xFF06」が示す命令４６は、命令「return」である。
【００７５】
つまり、図８に示すライブラリ「lib_a」ｐｙ－１１は、レジスタＧ３が保持する値を

移動した後、レジスタＧ０～Ｇ２が保持する値を順次、移動するプログラムである。また、図８に示すスタック領域３０は、攻撃者によって改ざんされており、先に取り出される順に、戻り先アドレス「0xFFF6」、「0xFF01」、「0xFD01」、「0xFFF8」を有する。

【００７６】

リターン命令（図示せず）を検出すると（図６のＳ１１のＹｅｓ）、分岐部１９は、スタック領域３０から戻り先アドレス「0xFFF6」を取得し、プログラムカウンタ１０に設定するとともに、戻り先チェック部１７を有効化する（Ｓ１２）。戻り先アドレス「0xFFF6」は、改ざんされたアドレスであって、ライブラリ「lib_b」ｐｙ－１２内の命令「mov G2, 0xffff」を示す。

【００７７】

次に、命令フェッチ部１２は、命令「mov G2, 0xffff」を取得し、命令デコード部１３に出力する。命令デコード部１３は、命令「mov G2, 0xffff」をデコードし（Ｓ１３）、戻り先チェック部１７が有効化されているか否かを判定する（Ｓ１４）。

【００７８】

工程Ｓ１２により、戻り先チェック部１７が有効化されている（Ｓ１４のＹｅｓ）。したがって、命令デコード部１３は、戻り先チェック部１７を無効化するとともに（Ｓ１５）、取得した命令「mov G2, 0xffff」が戻り先ラベルを有するか否かを判定する（Ｓ１６）。命令「mov G2, 0xffff」は、戻り先ラベルを有していないため（Ｓ１６のＮｏ）、戻り先チェック部１７は、例外割り込みを発生させる（Ｓ１９）。

【００７９】

図９は、図５に示した情報処理装置１００の処理を、図１に示したスタック領域３０が改ざんされている場合に基づいて説明する図（その２）である。図９において、図１で示したものと同一のものは、同一の記号で示す。

【００８０】

図９に示す、ライブラリ「lib_a」ｐｙ－１１、及び、ライブラリ「lib_b」ｐｙ－１２は、図８と同一である。ただし、図８には図示していないが、ライブラリ「lib_b」ｐｙ－１２のアドレス「0xFFF5」は、命令（ラベル）「LABEL_NON_RET」を有する。命令（ラベル）「LABEL_NON_RET」は、不適切なラベルである。また、図９に示すスタック領域３０は、図８に示すスタック領域３０と異なる。図９に示すスタック領域３０は、攻撃者によって改ざんされており、先に取り出される順に、戻り先アドレス「0xFFF5」、「0xFF01」、「0xFD01」、「0xFFF8」を有する。

【００８１】

リターン命令（図示せず）を検出すると（図６のＳ１１のＹｅｓ）、分岐部１９は、スタック領域３０から戻り先アドレス「0xFFF5」を取得し、プログラムカウンタ１０に設定するとともに、戻り先チェック部１７を有効化する（Ｓ１２）。戻り先アドレス「0xFFF5」は、改ざんされたアドレスであって、ライブラリ「lib_b」ｐｙ－１２内の命令（ラベル）「LABEL_NON_RET」を示す。

【００８２】

次に、命令フェッチ部１２は、命令（ラベル）「LABEL_NON_RET」を取得し、命令デコード部１３に出力する。命令デコード部１３は、命令（ラベル）「LABEL_NON_RET」をデコードし（Ｓ１３）、戻り先チェック部１７が有効化されているか否かを判定する（Ｓ１４）。

【００８３】

工程Ｓ１２により、戻り先チェック部１７が有効化されている（Ｓ１４のＹｅｓ）。したがって、命令デコード部１３は、戻り先チェック部１７を無効化するとともに（Ｓ１５）、取得した命令（ラベル）「LABEL_NON_RET」が戻り先ラベルを有するか否かを判定する（Ｓ１６）。図９の例では、戻り先ラベルを有するため（Ｓ１６のＹｅｓ）、戻り先ラベル検証部１８は、取得した命令（ラベル）「LABEL_NON_RET」が、適切なラベル「LABEL_RET」と一致するか否かを判定する（Ｓ１７、Ｓ１８）。取得した命令（ラベル）「LABEL_NON_RET」が適切なラベルと一致しないため（Ｓ１８のＮｏ）、戻り先ラベル検証部１

８は、例外割り込みを発生させる（Ｓ１９）。

【００８４】

図７～図９に示すように、第１の実施の形態における情報処理装置１００は、リターン命令の戻り先アドレスを判定することによって、ＲＯＰ攻撃を検出することができる。そして、情報処理装置１００は、ＲＯＰ攻撃を検出した場合に、例外割り込みを発生させることで、命令実行処理を停止することができる。また、第１の実施の形態によると、情報処理装置１００に対するハードウェアの変更箇所は、極めて少ない。したがって、少ないハードウェアの変更により、効率的にＲＯＰ攻撃を検出し、回避することが可能になる。

【００８５】

また、前述したとおり、第１の実施の形態における情報処理装置１００は、攻撃者がガジェットｇａとして使用可能な命令列を大幅に限定することができる。したがって、仮に、ＲＯＰの攻撃者が、図５に示した情報処理装置１００の構成を検知している場合であっても、攻撃者がガジェットｇａとして利用する命令列を限定することが可能になる。つまり、攻撃者が、本実施の形態における情報処理装置１００の構成を検知したところで、ガジェットｇａとして使用可能な命令列は少ない。

【００８６】

したがって、本実施の形態によると、情報処理装置１００の構成が攻撃者に検知されている場合であっても、ガジェットｇａに基づいて所望のプログラムを構成することを困難にし、ＲＯＰ攻撃の容易性を低くすることが可能になる。したがって、ＲＯＰ攻撃の実施を抑制することが可能になる。

【００８７】

[第２の実施の形態]

第２の実施の形態では、第１の実施の形態に加えて、情報処理装置がＲＯＰ攻撃の検出の対象とするプログラムを限定する。第２の実施の形態では、図３、図４に示したアセンブリコードｐｘ、ｐｙは、リターン命令「return」の代わりに、検証用リターン命令「return_verify」を有する。第２の実施の形態における情報処理装置は、検証用のリターン命令を検出した場合に、戻り先アドレスの判定処理を行う。

【００８８】

より具体的に、第２の実施の形態における情報処理装置２００は、リターン命令の処理時に、当該リターン命令が、検証用リターン命令「return_verify」であるか否かを判定する。情報処理装置２００は、リターン命令が検証用リターン命令である場合に、第１の実施の形態で説明したように、戻り先アドレスの命令が戻り先ラベルを有するか否かを判定する。

【００８９】

一方、リターン命令が検証用リターン命令ではない場合、情報処理装置２００は、戻り先アドレスの命令が戻り先ラベルを有するか否かを判定することなく、戻り先のルーチンの処理を続行する。

【００９０】

これにより、第２の実施の形態の情報処理装置は、一定のプログラムを、ＲＯＰ攻撃の検出の対象外とすることができる。つまり、一定のプログラムに対する、ＲＯＰ攻撃の検出処理を省略することが可能になる。これにより、検出処理の対象外とする一定のプログラムへの戻り先ラベルの付加を省略することが可能になる。

【００９１】

図１、図２で前述したとおり、ＲＯＰの攻撃者は、カーネル空間で動作するプログラムｐａに基づいたガジェットｇａを利用する。これにより、カーネル空間で、昇格した権限にしたがって疑似プログラムｐｂ（図１、図２）と同様の処理を実行させ、カーネルを破壊させる。一方、ユーザ空間で動作するプログラムは、ガジェットｇａとして利用され難い。したがって、情報処理装置は、ユーザ空間で動作するプログラムのＲＯＰ攻撃の検出処理を、省略してもよい。

【００９２】

（アセンブリコード）
図１０は、第２の実施の形態における、アセンブリコードｐｚの一例を示す図である。図１０に示すアセンブリコードｐｚは、図１、図２に示したプログラムｐａの一例であって、アセンブリコードで記述されたプログラムである。また、図１０に示すアセンブリコードｐｚは、図３に示したアセンブリコードｐｘに対応する。
【００９３】
図１０に示すアセンブリコードｐｚは、命令５１「call lib_b」、命令５２「mov G0, 0xffff, ret」、命令５３「mov G1, 0xffff」、命令５４「mov G2, 0xffff」、命令「…」、命令５ｎ「return_verify」を有する。命令５２「mov G0, 0xffff, ret」における値「ret」は、戻り先ラベルである。命令５ｎ「return_verify」に示す、命令「return_verify」は検証用リターン命令である。
【００９４】
図１０に示すアセンブリコードｐｚは、検証用リターン命令「return_verify」を有する。したがって、アセンブリコードｐｚは、ＲＯＰ攻撃の検出の対象のプログラムであって、カーネル空間で動作するプログラム（図１）ｐａである。また、第２の実施の形態例において、ユーザ空間で動作するプログラムは、検証用リターン命令「return_verify」、及び、戻り先ラベルを有しない。
【００９５】
（情報処理装置のハードウェア構成）
図１１は、第２の実施の形態における情報処理装置のハードウェア構成を示す図である。図１１において、図５で示したものと同一のものは、同一の記号で示す。図１１に示す情報処理装置２００は、図５に示す、第１の実施の形態の情報処理装置１００と同様にして、プログラムカウンタ１０、命令メモリ１１、命令フェッチ部１２、命令デコード部１３、命令実行部１４等を有する。
【００９６】
第２の実施の形態における情報処理装置２００は、第１の実施の形態における情報処理装置１００と、命令実行部１４が異なる。第２の実施の形態の命令実行部１４は、さらに、検証リターン命令実行部２０を有する。検証リターン命令実行部２０は、処理対象のリターン命令が、検証用リターン命令「return_verify」であるか否かを判定する。リターン命令が検証用リターン命令「return_verify」である場合、検証リターン命令実行部２０は、戻り先チェック部１７を有効化する。また、分岐部１９は、スタック領域（図１）３０から戻り先アドレスを取り出し、プログラムカウンタ１０に設定する。後続の処理は、第１の実施の形態における図６のフローチャート図で説明したとおりである。
【００９７】
一方、リターン命令が通常のリターン命令「return」である場合、検証リターン命令実行部２０は、リターン命令による分岐時であっても、戻り先チェック部１７を有効化しない。命令実行部１４の分岐部１９は、リターン命令による分岐時に、スタック領域（図１）３０から戻り先アドレスを取り出し、プログラムカウンタ１０に設定する。検証リターン命令実行部２０は、分岐部１９による戻り先チェック部１７の有効化を抑止する。
【００９８】
次に、命令フェッチ部１２は、プログラムカウンタ１０が示す命令を、命令メモリ１１から読み出し、命令デコード部１３は、命令フェッチ部１２によって取得された命令をデコードする（図６のＳ１３）。戻り先チェック部１７が無効であるため（Ｓ１４のＮｏ）、命令デコード部１３は、デコードした命令を命令実行部１４に出力する。
【００９９】
図１０、図１１に示すように、第２の実施の形態における情報処理装置２００は、検証用リターン命令「return_verify」を設けることによって、一部のプログラムをＲＯＰ攻撃の検出の対象外とすることができる。
【０１００】
ユーザ空間で動作するプログラムは、様々な作成者によって作成され得る。したがって

、すべての作成者に対して、プログラムへの戻り先ラベルの付加を要求することは容易ではない。また、ユーザ空間で動作するプログラムは、様々なコンパイラによって生成され得る。すべてのコンパイラに、図３に示した戻り先ラベルのフィールドの付加機能を追加することも容易ではない。

【０１０１】

このように、ユーザ空間で動作するすべてのプログラムに、戻り先ラベルを付加することは容易ではない。情報処理装置１００が、戻り先ラベルが付加されていないプログラムを実行した場合、リターン命令の実行時に例外割り込みが発生してしまう。

【０１０２】

したがって、第２の実施の形態では、カーネル空間で動作するプログラムのみを対象としてＲＯＰ攻撃を検出する。即ち、第２の実施の形態では、戻り先ラベルを付加するプログラムを、カーネル空間で動作するプログラムｐａ（図１）に限定する。これにより、プログラムへの変更範囲が最小限になることから、より効率的に、ＲＯＰ攻撃を検出することが可能になる。また、本実施の形態における命令実行制御方法を適用することによる、ユーザへの負荷も抑えることが可能になる。

【０１０３】

なお、図１０では、検証用リターン命令が、命令「return_verify」である場合を例示した。ただし、この例に限定されるものではない。検証用リターン命令は、通常のリターン命令「return」に、所定のパラメータが付加された形態であってもよい。

【０１０４】

［他の実施の形態］

第１の実施の形態では、戻り先ラベルが、予め定めた値（「LABEL_RET」）である場合を例示した。ただし、戻り先ラベルは、メインルーチンが呼び出したサブルーチンの種別に応じて値が異なっていてもよい。例えば、図３、図４に示したアセンブリコードｐｘ、ｐｙの戻り先ラベルは、コール命令によって呼び出したサブルーチンに応じて、異なる値を有する。

【０１０５】

したがって、戻り先ラベル検証部１８は、例えば、戻り先ラベル（コール命令によって呼び出したサブルーチンに応じた値）が、リターン命令が属するサブルーチンに応じたラベル値と一致するか否かを判定する（図６のＳ１８）。

【０１０６】

一致する場合（Ｓ１８のＹｅｓ）、コール命令によって呼び出したサブルーチンと、リターン命令が属するサブルーチンとが一致することを示し、ルーチン間の遷移が適切であることを示す。ルーチン間の遷移が適切である場合、スタック領域３０が改ざんされていないことを示すため、命令デコード部１３は、デコードした命令を命令実行部１４に出力する。

【０１０７】

一方、一致しない場合（Ｓ１８のＮｏ）、コール命令によって呼び出したサブルーチンと、リターン命令が属するサブルーチンとが異なることを示し、ルーチン間の遷移が不正であることを示す。ルーチン間の遷移が不正である場合、スタック領域３０が改ざんされていることを示すため、戻り先ラベル検証部１８は、例外割り込みを発生させる。

【０１０８】

このように、他の実施の形態によると、戻り先ラベルの値に基づいて、ルーチン間の遷移が適切か否かを判定可能になる。これにより、スタック領域３０の改ざんを検出可能になる。したがって、ＲＯＰの攻撃者が、仮に、第１、第２の実施の形態における情報処理装置１００、２００の構成を検知している場合であっても、ＲＯＰ攻撃の容易性をより低くすることができ、ＲＯＰ攻撃の実施を抑制することが可能になる。

【０１０９】

以上の実施の形態をまとめると、次の付記のとおりである。

【０１１０】

（付記１）
コール命令を含む第１の命令群と、リターン命令を含む第２の命令群とを記憶する記憶部と、
前記第１の命令群から前記コール命令によって呼び出された前記第２の命令群の実行時に、前記第２の命令群の前記リターン命令による、前記第１の命令群への戻り先アドレスの命令が識別情報を有するか否かを判定し、前記識別情報を有する場合に、前記第１の命令群の処理を続行し、前記識別情報を有しない場合に、命令実行処理を停止する処理部、とを有する、
命令実行制御装置。
【０１１１】
（付記２）
付記１において、
前記処理部は、前記戻り先アドレスの命令に前記識別情報が付加されているか否かに基づいて、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御装置。
【０１１２】
（付記３）
付記１において、
前記処理部は、前記戻り先アドレスの命令が前記識別情報であるか否かに基づいて、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御装置。
【０１１３】
（付記４）
付記１乃至３のいずれかにおいて、
前記処理部は、さらに、前記戻り先アドレスの命令が前記識別情報を有する場合に前記識別情報が所定の情報であるか否かを判定し、前記識別情報が前記所定の情報である場合、前記第１の命令群の処理を続行し、前記所定の情報ではない場合、前記命令実行処理を停止する、
命令実行制御装置。
【０１１４】
（付記５）
付記１乃至４のいずれかにおいて、
前記処理部は、さらに、前記第２の命令群の前記リターン命令が検証用のリターン命令であるか否かを判定し、前記検証用のリターン命令である場合に、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定し、前記検証用のリターン命令ではない場合に、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定せず前記第１の命令群の処理を続行する、
命令実行制御装置。
【０１１５】
（付記６）
付記１乃至５のいずれかにおいて、
前記処理部は、前記識別情報を有しない場合に、例外割り込みを発生させる、
命令実行制御装置。
【０１１６】
（付記７）
付記１乃至６のいずれかにおいて、
前記記憶部はスタック領域を含み、
前記処理部は、前記第２の命令群の前記リターン命令による前記第１の命令群への戻り先アドレスを前記スタック領域から取得し、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、

命令実行制御装置。
【０１１７】
（付記８）
付記１乃至７のいずれかにおいて、
前記処理部は、
判定部を含み命令をデコードする命令デコード部と、分岐部を含み前記命令を実行する命令実行部とを有し、
前記分岐部は、前記命令実行部による前記第２の命令群の前記リターン命令の実行時に、前記戻り先アドレスの命令の判定を前記判定部に指示し、
前記判定部は、前記命令デコード部がデコードした前記戻り先アドレスの命令が、前記識別情報を有するか否かを判定する、
命令実行制御装置。
【０１１８】
（付記９）
第１の命令群のコール命令によって呼び出された、リターン命令を含む前記第２の命令群の実行時に、前記第２の命令群の前記リターン命令による、前記第１の命令群への戻り先アドレスの命令が識別情報を有するか否かを判定し、
前記識別情報を有する場合に、前記第１の命令群の処理を続行し、前記識別情報を有しない場合に、命令実行処理を停止する、
命令実行制御方法。
【０１１９】
（付記１０）
付記９において、
前記判定は、前記戻り先アドレスの命令に前記識別情報が付加されているか否かに基づいて、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御方法。
【０１２０】
（付記１１）
付記９において、
前記判定は、前記戻り先アドレスの命令が前記識別情報であるか否かに基づいて、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御方法。
【０１２１】
（付記１２）
付記９乃至１１のいずれかにおいて、
前記判定は、さらに、前記戻り先アドレスの命令が前記識別情報を有する場合に前記識別情報が所定の情報であるか否かを判定し、
前記停止は、前記識別情報が前記所定の情報である場合に前記第１の命令群の処理を続行し、前記所定の情報ではない場合に前記命令実行処理を停止する、
命令実行制御方法。
【０１２２】
（付記１３）
付記９乃至１２のいずれかにおいて、
前記判定は、さらに、前記第２の命令群の前記リターン命令が検証用のリターン命令であるか否かを判定し、前記検証用のリターン命令である場合に、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定し、前記検証用のリターン命令ではない場合に、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定せず前記第１の命令群の処理を続行する、
命令実行制御方法。
【０１２３】

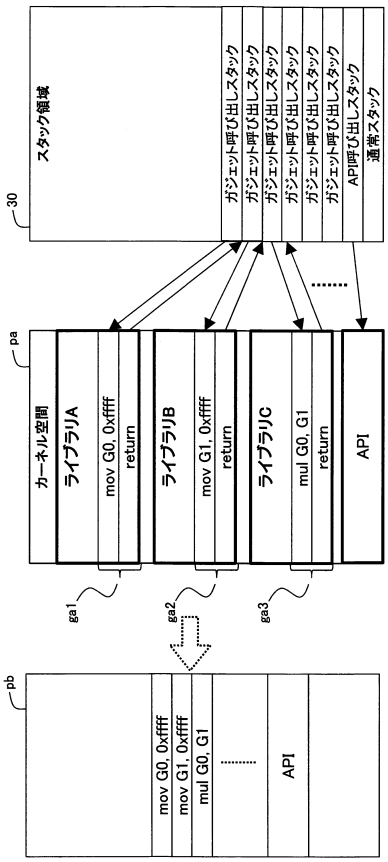
（付記１４）
付記９乃至１３のいずれかにおいて、
前記停止は、前記識別情報を有しない場合に、例外割り込みを発生させる、
命令実行制御方法。

【０１２４】
（付記１５）
付記９乃至１４のいずれかにおいて、
前記判定は、前記第２の命令群の前記リターン命令による前記第１の命令群への戻り先アドレスをスタック領域から取得し、前記戻り先アドレスの命令が前記識別情報を有するか否かを判定する、
命令実行制御方法。

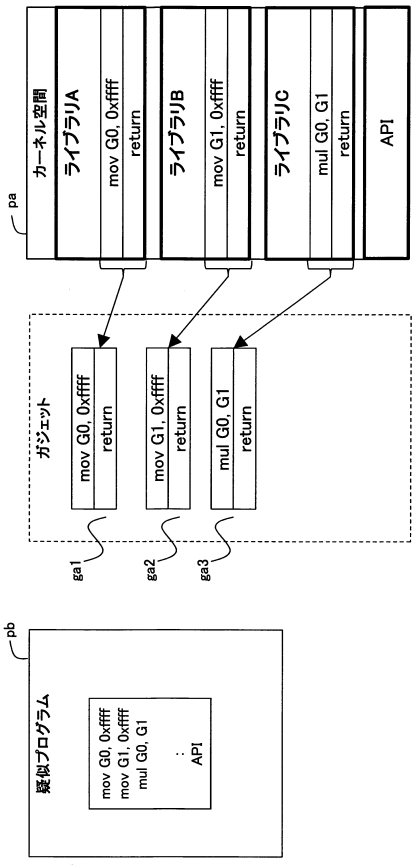
【符号の説明】

【０１２５】
１００：情報処理装置、１０プログラムカウンタ、１１：命令メモリ、１２：命令フェッチ部、１３：命令デコード部、１４：命令実行部

【図１】



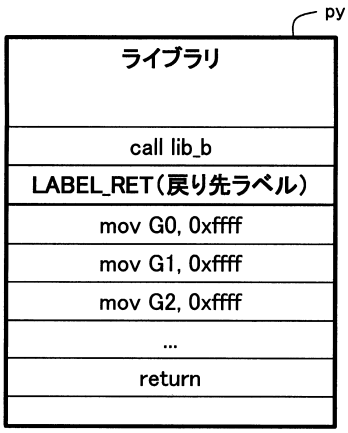
【図２】



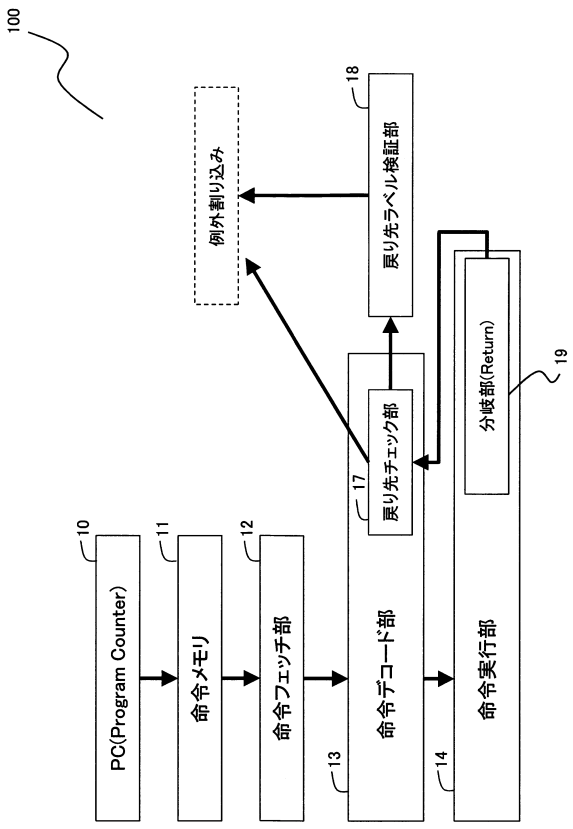
【図３】


px
ライブラリ
call lib_b
mov G0, 0xffff, ret
mov G1, 0xffff
mov G2, 0xffff
...
return

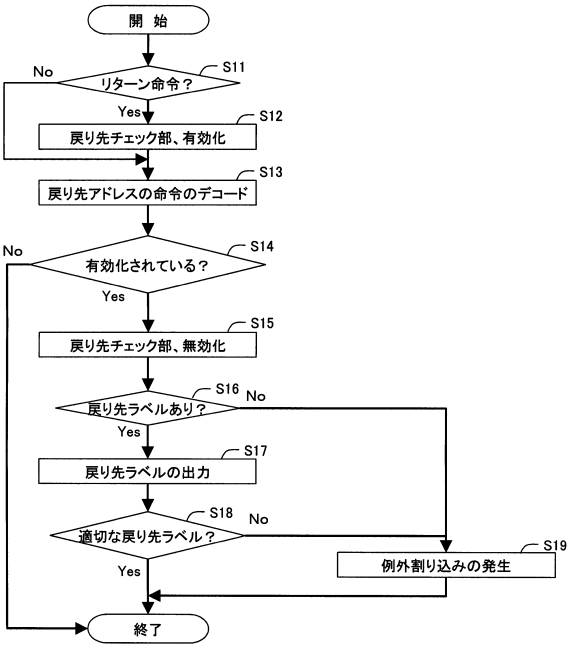
【図４】


py
ライブラリ
call lib_b
LABEL_RET（戻り先ラベル）
mov G0, 0xffff
mov G1, 0xffff
mov G2, 0xffff
...
return

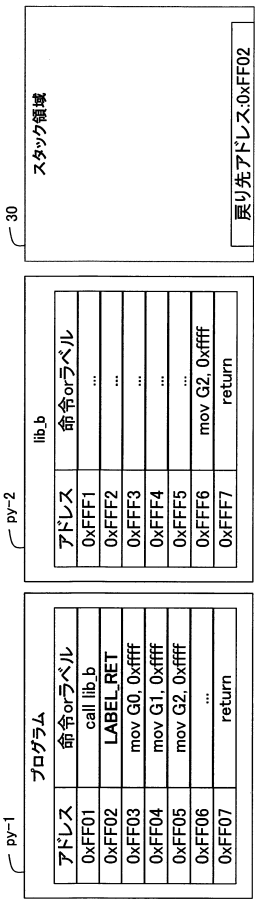
【図５】


100
例外割り込み
18
戻り先ラベル検証部
19
分岐部(Return)
17
戻り先チェック部
10
PC(Program Counter)
11
命令メモリ
12
命令フェッチ部
13
命令デコード部
14
命令実行部

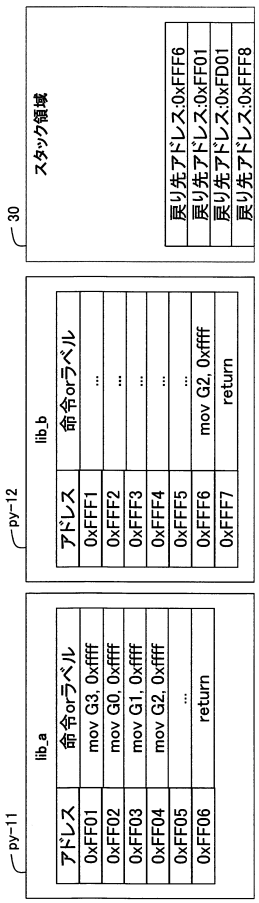
【図６】


開 始
S11
リターン命令？
No
Yes
S12
戻り先チェック部、有効化
S13
戻り先アドレスの命令のデコード
S14
有効化されている？
No
Yes
S15
戻り先チェック部、無効化
S16
戻り先ラベルあり？
No
Yes
S17
戻り先ラベルの出力
S18
適切な戻り先ラベル？
No
Yes
S19
例外割り込みの発生
終了

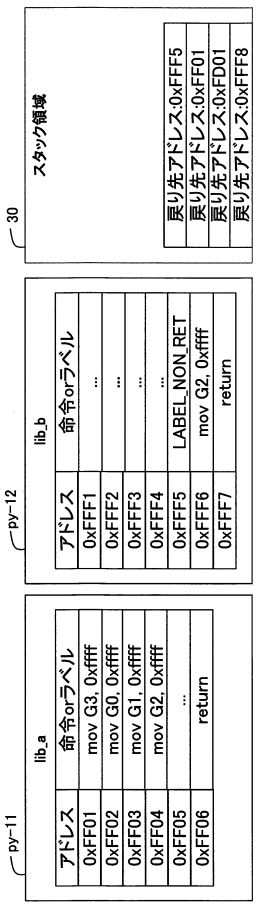
【図７】


py-1
プログラム
アドレス
命令orラベル
0xFF01
call lib_b
0xFF02
LABEL_RET
0xFF03
mov G0, 0xffff
0xFF04
mov G1, 0xffff
0xFF05
mov G2, 0xffff
0xFF06
...
0xFF07
return
py-2
lib_b
アドレス
命令orラベル
0xFFF1
...
0xFFF2
...
0xFFF3
...
0xFFF4
...
0xFFF5
...
0xFFF6
mov G2, 0xffff
0xFFF7
return
30
スタック領域
戻り先アドレス:0xFF02

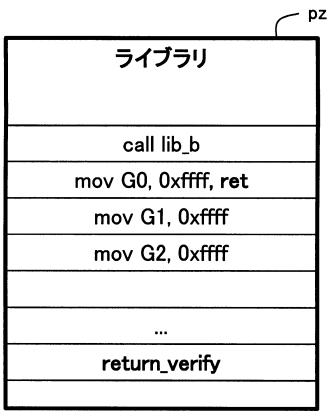
【図８】


py-11
lib_a
アドレス
命令orラベル
0xFF01
mov G3, 0xffff
0xFF02
mov G0, 0xffff
0xFF03
mov G1, 0xffff
0xFF04
mov G2, 0xffff
0xFF05
...
0xFF06
return
py-12
lib_b
アドレス
命令orラベル
0xFFF1
...
0xFFF2
...
0xFFF3
...
0xFFF4
...
0xFFF5
...
0xFFF6
mov G2, 0xffff
0xFFF7
return
30
スタック領域
戻り先アドレス:0xFFF6
戻り先アドレス:0xFF01
戻り先アドレス:0xFD01
戻り先アドレス:0xFFF8

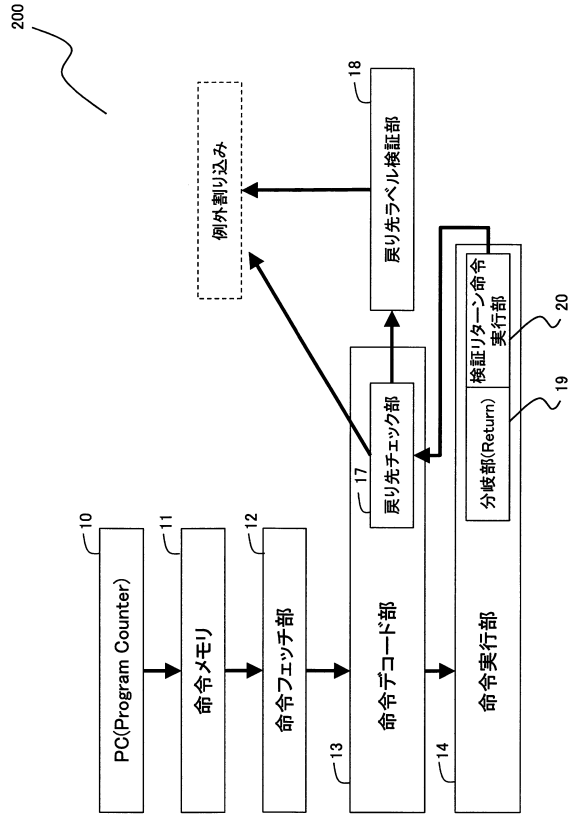
【図９】


py-11
lib_a
アドレス
命令orラベル
0xFF01
mov G3, 0xffff
0xFF02
mov G0, 0xffff
0xFF03
mov G1, 0xffff
0xFF04
mov G2, 0xffff
0xFF05
...
0xFF06
return
py-12
lib_b
アドレス
命令orラベル
0xFFF1
...
0xFFF2
...
0xFFF3
...
0xFFF4
...
0xFFF5
LABEL_NON_RET
0xFFF6
mov G2, 0xffff
0xFFF7
return
30
スタック領域
戻り先アドレス:0xFFF5
戻り先アドレス:0xFF01
戻り先アドレス:0xFD01
戻り先アドレス:0xFFF8

【図１０】


pz
ライブラリ
call lib_b
mov G0, 0xffff, ret
mov G1, 0xffff
mov G2, 0xffff
...
return_verify

【図１１】


200
例外割り込み
18
戻り先ラベル検証部
20
検証リターン命令実行部
19
分岐部(Return)
17
戻り先チェック部
10
PC(Program Counter)
11
命令メモリ
12
命令フェッチ部
13
命令デコード部
14
命令実行部

フロントページの続き

(72)発明者　武仲　正彦
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

審査官　宮司　卓佳

(56)参考文献　特開２０１５－１９１６５８（ＪＰ，Ａ）
特開２０１１－１２３６５８（ＪＰ，Ａ）
米国特許出願公開第２０１４／００９６２４５（ＵＳ，Ａ１）
米国特許第０７５５２４７７（ＵＳ，Ｂ１）
特開平０５－２１６７２１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　２１／５４
Ｇ０６Ｆ　９／３２