(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2006-106939
(P2006-106939A)
(43)公開日　平成18年4月20日(2006.4.20)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 21/22	(2006.01)	G06F	9/06	660N	5B042
G06F 11/28	(2006.01)	G06F	11/28	310E	5B076
		G06F	11/28	320C	5B276

審査請求 未請求　請求項の数 7　OL　（全 16 頁）

(21)出願番号　特願2004-290053 (P2004-290053)
(22)出願日　平成16年10月1日 (2004.10.1)

(71)出願人　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　110000198
特許業務法人湘洋内外特許事務所
(72)発明者　西山　博泰
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社日立製作所システム開発研究所内

Fターム(参考)　5B042 GA21 HH30 JJ10 KK01 KK13
LA10 MA08 MC04 MC15
5B076 FD08
5B276 FD08

(54)【発明の名称】侵入検知方法及び侵入検知装置並びにプログラム

(57)【要約】
【課題】　コンピュータへの不正プログラムの進入を検知する侵入検知方法及び侵入検知装置並びにプログラムを提供する。
【解決手段】　予め監視対象プログラムを静的に解析し、システムコールと、当該システムコールまでの経路情報とを対応付けて記憶する。監視対象プログラムを実行し、システムコールを要求すると、当該システムコール及び当該システムコールまでの経路情報が、予め解析したシステムコール及び当該システムコールに対応する経路情報に一致するか否か判定する。一致しない場合、不正侵入があったと判定し、システム管理者への通知、監視対象プログラムの停止やスローダウン等を指示する。
【選択図】　図１

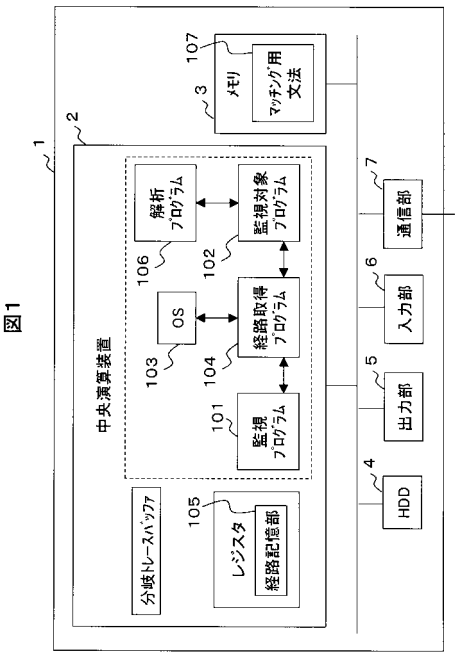


図1

【特許請求の範囲】
【請求項１】
コンピュータが、監視対象プログラムへの不正侵入を検知する侵入検知方法であって、
前記監視対象プログラムへの不正侵入がない場合のシステムコールと、当該システムコールまでの経路情報とを対応付けて記憶部へ記憶する第１のステップと、
前記監視対象プログラムの実行によりシステムコールを要求する場合、当該システムコールを示す情報と、当該システムコールまでの経路情報とを取得する第２のステップと、
前記要求したシステムコールを示す情報と当該システムコールまでの経路情報とが、前記記憶したシステムコールと当該システムコールに対応する経路情報とに一致するか否か判定する第３のステップと、
前記第３のステップで一致すると判定した場合、前記要求したシステムコールを実行する第４のステップと、
を備えることを特徴とする侵入検知方法。
【請求項２】
請求項１記載の侵入検知方法において、システムコールまでの経路情報とは、監視対象プログラム内の経路、又は、監視対象プログラムを実行するコンピュータのメモリ領域の経路であることを特徴とする侵入検知方法。
【請求項３】
請求項１記載の侵入検知方法において、システムコールまでの経路情報とは、分岐アドレスであることを特徴とする侵入検知方法。
【請求項４】
請求項３記載の侵入検知方法において、分岐アドレスは、分岐トレースバッファの機能により取得することを特徴とする侵入検知方法。
【請求項５】
請求項１乃至４記載の侵入検知方法において、前記第３のステップで一致しないと判定した場合、出力手段への侵入警告の出力、前記監視対象プログラムの実行停止、前記監視対象プログラムのスローダウンのうち少なくとも一つを実行する第５のステップと
をさらに備えることを特徴とする侵入検知方法。
【請求項６】
コンピュータが、監視対象プログラムへの不正侵入を検知する侵入検知装置であって、
前記監視対象プログラムへの不正侵入がない場合のシステムコールと、当該システムコールまでの経路情報とを対応付けて記憶する記憶手段と、
前記監視対象プログラムの実行によりシステムコールを要求する場合、当該システムコールを示す情報と、当該システムコールまでの経路情報とを取得する取得手段と、
前記要求したシステムコールを示す情報と当該システムコールまでの経路情報とが、前記記憶したシステムコールと当該システムコールに対応する経路情報とに一致するか否か判定する判定手段と、
前記判定手段により一致すると判定した場合、前記要求したシステムコールを実行する実行手段と、
を備えることを特徴とする侵入検知装置。
【請求項７】
監視対象プログラムへの不正侵入を検知する侵入検知プログラムであって、
コンピュータが、
前記監視対象プログラムへの不正侵入がない場合のシステムコールと、当該システムコールまでの経路情報とを対応付けて記憶部へ記憶する第１のステップと、
前記監視対象プログラムの実行によりシステムコールを要求する場合、当該システムコールを示す情報と、当該システムコールまでの経路情報とを取得する第２のステップと、
前記要求したシステムコールを示す情報と当該システムコールまでの経路情報とが、前記記憶したシステムコールと当該システムコールに対応する経路情報とに一致するか否か判定する第３のステップと、

前記第３のステップで一致すると判定した場合、前記要求したシステムコールを実行する第４のステップと、
をコンピュータに実行させることを特徴とする侵入検知プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、コンピュータへの不正プログラムの進入を検知する侵入検知方法及び侵入検知装置並びにプログラムに関するものである。
【背景技術】
【０００２】
コンピュータシステムのセキュリティホールを攻撃して侵入するウイルスやワーム等の不正プログラムは、侵入したコンピュータの制御を奪い、コンピュータそのものや当該コンピュータに格納されているデータの破壊、他コンピュータへの不正なデータの送信等を行う。このような不正プログラムに対応するため、プログラムやシステムの挙動を監視して、コンピュータの制御が奪われたか否かを検知する侵入検知システム（ＩＤＳ：Intrusion Detection System）がある。
【０００３】
侵入検知の例として、Wagnerらによって提唱された、不正プログラムによるプログラムの実行制御の異常を、プログラムの静的解析と外部からのプログラム実行状態の監視により確認する方法がある（非特許文献１）。
【０００４】
この方法では、予め、監視対象プログラムの制御フローを解析して、当該プログラムの発行するシステムコール（system call）の発行シーケンスを解析しておく。システムコールとは、ＯＳ（Operating System）の機能を、ＯＳ上で動作するアプリケーションプログラムで呼び出すことである。監視対象プログラム実行時に、システムコールの発行過程を外部から監視し、その発行シーケンスが、予め解析したシステムコール発行シーケンスと合致するかを判定することにより、当該プログラムが不正な動作を行ったかどうか確認する。
【０００５】
Wagnerらの方法を具体的に説明する。図１０は、Wagnerらによる侵入検知の方法を説明する図である。図１０（ａ）は、監視対象プログラム１００１の制御フローを示している。ここで、丸で囲んだ要素は、監視対象プログラムのシステムコールの発行を、矢印はシステムコール間の制御の遷移を表している。
【０００６】
図１０（ｂ）は、図１０（ａ）に示す監視対象プログラム１００１のシステムコール発行シーケンスを示す。ここでは、監視対象プログラムのシステムコール発行シーケンスを文脈自由な文法形式で表したものをマッチング用文法として説明する。図１０（ｂ）は、監視対象プログラム１００１のマッチング用文法１００２の一例である。ここで、「Entry」はプログラムの開始点を示し、「＊」は０回以上の繰り返しを示す。
【０００７】
マッチング用文法１００２は、「open」システムコールを発行した後に、「read」システムコールを０回以上繰り返し発行し、その後、「close」、「exec」、「exit」の順でシステムコールを発行することを示している。
【０００８】
図１０（c）は、Wagnerらの方法による侵入検知システムの概要を示している。Wagnerらの方法による侵入検知システムは、監視対象プログラム１００１、マッチング用文法１００２、監視プログラム１００３等を備える。
【０００９】
予め、監視対象プログラム１００１を解析し、マッチング用文法１００２を生成しておく。監視対象プログラム１００１は、システムコールを発行する場合、当該システムコー

ルを示す情報を監視プログラム１００３に送信する。監視プログラム１００３は、マッチング用文法１００２を参照し、監視対象プログラム１００１のシステムコール発行シーケンスが、マッチング用文法１００２に示すシステムコール発行シーケンスと一致するか否か判定する。システムコールシーケンスが一致しない場合は、外部からの侵入によってプログラムの制御が奪われた可能性があるので、監視プログラム１００３は、監視対象プログラム１００１の停止、スローダウン、システム管理者への通知を行う。

【００１０】

【非特許文献１】D.Wagner and D.Dean. Intrusion Detection via Static Analysis,2001 IEEE Symposium on Security and Privacy,IEEE,2001.

【発明の開示】

【発明が解決しようとする課題】

【００１１】

上述したWagnerらの侵入検知方法では、不正プログラムの攻撃により監視対象プログラムの制御が奪われた後のシステムコールの発行シーケンスが、元のシステムコール発行シーケンスと一致する場合、不正侵入があったか否か検知することができない。

【００１２】

不正侵入を検知できない場合の例を図１１を参照して説明する。図１１（ａ）において、監視対象プログラム１１０１は、関数「main」から関数「foo」を呼び出している。関数「foo」の中で、大きさ１０バイトの配列「buf」を宣言し、システムコール「gets」により、配列「buf」にデータを読み込んでいる。「gets」を呼び出した後、「return」で関数「main」に戻り、システムコール「exec」、「exit」を発行している。

【００１３】

監視対象プログラム１１０１の、Wagnerらの方法によるマッチング用文法の一例を、図１１（ｂ）のマッチング用文法１１０２に示す。マッチング用文法１１０２に示すように、監視対象プログラム１１０１はシステムコールを「gets」、「exec」、「exit」の順に発行する。

【００１４】

図１１（ｃ）は、監視対象プログラム１１０１が不正プログラムに攻撃される前、即ち、監視対象プログラム１１０１が正常に実行された場合のスタックフレームの状態の一例を示す。配列「buf」１１０３に対応する領域は、関数「foo」のアクティベーションレコード中に確保されている。宣言した配列「buf」１１０３の下の「PC」１１０４には、関数「foo」の戻りアドレス「main+m1」が格納されている。

【００１５】

監視対象プログラム１１０１は次のように動作する。監視対象プログラム１１０１は、関数「main」から呼び出した関数「foo」の中で「gets」により配列「buf」にデータを読み込んだ後、「return」で「PC」１１０４を参照し、関数「foo」の戻りアドレス「main+m1」を得て、関数「main」に戻り、システムコール「exec」、「exit」を実行する。

【００１６】

図１１（ｄ）は、不正プログラムが実行された後に監視対象プログラム１１０１が実行された場合のスタックフレームの状態の一例を示す。配列「buf」１１０３に対応する領域は、関数「foo」のアクティベーションレコード中に確保されている。

【００１７】

監視対象プログラム１１０１は次のように動作する。監視対象プログラム１１０１は、関数「main」から呼び出した関数「foo」の中で、システムコール「gets」により配列「buf」にデータを読み込む。ここまでは、監視対象プログラム１１０１が正常に実行された場合と同じである。

【００１８】

不正プログラムによる攻撃を受けたことにより、「gets」の呼び出しで、配列「buf」１１０３の領域長１０バイトを超えた不正なデータが読み込まれる。すると、本来正しい戻りアドレスが格納されているはずの「PC」１１０４に、「gets」の呼び出しにより読み

込まれた不正なデータが書き込まれ、関数「foo」の戻りアドレスが、攻撃コード１１０５のアドレスを示すように変更される。また、不正プログラムにより、スタック上に攻撃コード１１０５が書き込まれる。当該アドレスに従って戻った先の不正プログラムのシステムコールが「exec」であれば、不正なプログラムは実行される。

【００１９】

Wagnerらの侵入検知方法では、システムコールの発行シーケンスのマッチングにより不正侵入を検知している。従って、上述したような、システムコールの発行シーケンスが、元のプログラムと、不正プログラムにより攻撃された後のプログラムとで一致する場合、不正侵入を検知することができない。

【００２０】

本発明はこのような事情に鑑みてなされたもので、コンピュータへの不正侵入を検知する侵入検知方法及び侵入検知装置並びにプログラム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】

【００２１】

本発明は上記の目的を達成するためになされたもので、監視対象プログラムへの不正侵入がない場合のシステムコール及び当該システムコールまでの経路情報を、監視対象プログラムの実行によるシステムコール要求及び当該システムコールまでの経路情報と比較することにより、当該プログラムを実行するコンピュータが不正侵入されたか否か判定するものである。

【００２２】

具体的には、例えば、コンピュータが、監視対象プログラムへの不正侵入を検知する侵入検知方法であって、前記監視対象プログラムへの不正侵入がない場合のシステムコールと、当該システムコールまでの経路情報とを対応付けて記憶部へ記憶する第１のステップと、前記監視対象プログラムの実行によりシステムコールを要求する場合、当該システムコールを示す情報と、当該システムコールまでの経路情報とを取得する第２のステップと、前記要求したシステムコールを示す情報と当該システムコールまでの経路情報とが、前記記憶したシステムコールと当該システムコールに対応する経路情報とに一致するか否か判定する第３のステップと、前記第３のステップで一致すると判定した場合、前記要求したシステムコールを実行する第４のステップと、を備えることを特徴とする。

【００２３】

また、例えば、システムコールまでの経路情報とは、監視対象プログラム内の経路、又は、監視対象プログラムを実行するコンピュータのメモリ領域の経路であることを特徴とする。

【００２４】

また、例えば、システムコールまでの経路情報とは、分岐アドレスであることを特徴とする。

【発明の効果】

【００２５】

本発明による情報処理装置及びプログラムによれば、コンピュータへの不正侵入を検知することができる。特に、不正侵入されたコンピュータが不正なプログラムにより攻撃され、自身で動作するプログラムの制御が奪われることを検知することができる。これにより、不正侵入されたコンピュータで実行するプログラムの停止、スローダウン、システム管理者への通知等の対応が可能となる。

【発明を実施するための最良の形態】

【００２６】

以下、本発明の一実施形態を図面を参照して詳細に説明する。図１は、本実施形態の侵入検知装置の構成の一例を示す図である。図１において、侵入検知装置１は、中央演算装置（ＣＰＵ：Central Processing Unit）２、メモリ３、ＨＤＤ（Hard Disk Drive）４、出力部５、入力部６、通信部７等を備える。

【００２７】

侵入検知装置１は、例えば計算機システム等である。本実施形態では、侵入検知装置１は、監視対象プログラム１０２を実行し、さらに、監視対象プログラム１０２を監視する監視プログラム１０１を実行するものとする。監視対象プログラム１０２を実行する侵入検知装置１は特定の機能の装置に限定されるものではないが、例えば、Ｗｅｂサーバやネットワークファイルサーバなど、通信ネットワークを介して他装置と通信する装置に適用するのが好適である。

【００２８】

中央演算装置２は、監視プログラム１０１、監視対象プログラム１０２、ＯＳ１０３、経路取得プログラム１０４をメモリ３にロードして実行するものとする。

【００２９】

監視プログラム１０１は、監視対象プログラム１０２を監視するプログラムである。具体的には、監視プログラム１０１は、監視対象プログラム１０２の実行により要求するシステムコール及び当該システムコールまでの経路情報と、監視対象プログラム１０２が不正な侵入を受けていない場合のシステムコール及び当該システムコールに対応する経路情報とが一致するか否かにより、監視対象プログラム１０２を監視する。本実施形態の侵入検知装置１では、中央演算装置２は、ＯＳ１０３を実行した上で、監視対象プログラム１０２を実行するものとする。

【００３０】

経路取得プログラム１０４とは、監視対象プログラム１０２を実行した場合、当該プログラムのシステムコールまでの経路情報を取得するプログラムである。システムコールまでの経路とは、例えば、監視対象プログラム１０２内の経路、又は、管理対象プログラム１０３を実行するメモリ３の領域の経路等である。

【００３１】

ここでは、システムコールまでの経路とは、監視対象プログラム１０２内の経路であるものとし、具体的には、監視対象プログラム１０２の分岐アドレスであるものとする。この分岐アドレスは、例えば、分岐トレースバッファの機能を用いることにより取得できる。分岐トレースバッファとは、一部の中央演算装置に備えられている機能であり、プログラムの実行により発生した分岐に関する情報を、中央演算装置上のレジスタ又はメモリ上の領域に記憶する機能である。

【００３２】

本実施形態では、経路取得プログラム１０４に、分岐トレースバッファの機能を用いるのとする。経路取得プログラム１０４は、取得した経路情報を、分岐トレースバッファの機能により取得した情報を、中央演算装置２上のレジスタに格納するものとする。以下、分岐トレースバッファの機能により取得した情報を格納するレジスタの領域を経路記憶部１０５として説明する。

【００３３】

経路取得プログラム１０４は、経路記憶部１０５に、図２に一例を示すような分岐に関する情報２０１を複数格納する。図２において、分岐に関する情報２０１には、分岐元アドレス、分岐先アドレス、分岐命令の分岐予測結果が含まれる。この経路記憶部１０５に格納する分岐に関する情報の個数は、利用者が入力部６を用いて指定することにより変更することができる。中央演算装置２は、ＯＳ１０３の実行等により、経路記憶部１０５へ格納する分岐に関する情報の個数を、特定のアクセス権を持つ利用者にのみ設定可能とする。

【００３４】

なお、分岐トレースバッファに関しては、例えば、“Intel,IA-32 Intel Architecture Software Developer's Manual Volume3:System Programming Guide,2003.”に記載されている。

【００３５】

図１において、解析プログラム１０６とは、監視対象プログラム１０２を静的に解析し、当該プログラムのシステムコールと、当該システムコールまでの経路情報とを取得する

ためのプログラムである。中央演算装置２は、解析プログラム１０６を実行し、監視対象プログラム１０２のシステムコールと、当該システムコールまでの経路情報とを取得し、取得したシステムコールと当該システムコールまでの経路情報とを対応付けてメモリ３に格納する。ここでは、中央演算装置２は、システムコールを示す情報と、当該システムコールまでの経路情報とを任意の文法表現で示したものをメモリ３に格納するものとし、以下、これをマッチング用文法として説明する。

【００３６】

システムコールまでの経路情報が分岐アドレスである場合のマッチング用文法の例を、図３を参照して説明する。図３において、マッチング用文法３０１は、中央演算装置２が、解析プログラム１０６を実行することにより、図１０（ａ）の監視対象プログラム１００１から生成した例である。

【００３７】

マッチング用文法３０１の「read.trace in {{0x2000,0x2001}}」は、「read」というシステムコールが「0x2000」、「0x2001」という順で分岐アドレスを経由することを示している。また、「open.trace in {{0x1000,0x1002},{0x1010,0x1002}}」は、「open」というシステムコールが、「{0x1000,0x1002}」又は「{0x1010,0x1002}」いずれかの分岐アドレスを経由することを示している。

【００３８】

図１において、メモリ３は、マッチング用文法１０７を記憶する。

【００３９】

出力部５は、例えば、ディスプレイ、スピーカ、プリンタ等の出力装置と、当該出力装置のドライバ等の制御部を備えるものである。

【００４０】

入力部６は、例えば、キーボード、マウス、マイク等の入力装置と、当該入力装置のドライバ等の制御部を備えるものである。

【００４１】

通信部７は、例えば、ＬＡＮ（Local Area Network）ポート等の通信インタフェースと、当該インタフェースの制御部を備えるものである。侵入検知装置１は、ＬＡＮを介するなどして、インターネットや他装置等（いずれも図示略）に接続されていてもよい。

【００４２】

なお、上述の実施形態において、図１を例とする侵入検出装置１の中央演算装置２内の各部及びマッチング用文法を、同一装置の中央演算装置及びメモリが備えるものとしたが、これに限られるわけではなく、各部の一部を他の装置の中央演算装置及びメモリが備えてもよい。例えば、第１の装置の中央演算装置は、監視対象プログラム１０２、経路取得プログラム１０４、経路記憶部１０５を備え、第１の装置と通信ネットワークを介して接続される第２の装置の中央演算装置は、解析プログラム１０６及び監視プログラム１０１、第２の装置のメモリがマッチング用文法１０７を備えてもよい。

【００４３】

次に、侵入検知装置１の動作を説明する。侵入検知装置１の中央演算装置２は、予め、解析プログラム１０６を実行して、マッチング用文法１０７を生成しておく。

【００４４】

なお、解析プログラム１０６の実行は、システム管理者の実行要求によるものや監視対象プログラム１０２の侵入検知装置１へのインストール時など、監視対象プログラム１０２が不正プログラムにより攻撃される前であればよく、特に限定するものではない。

【００４５】

マッチング用文法１０７を生成する動作を、図４を参照して説明する。なお、図４を参照して説明する動作は、中央演算装置２が解析プログラム１０６をメモリ３にロードして実行することにより実現する動作である。

【００４６】

中央演算装置２は、解析プログラム１０６を実行して、監視対象プログラム１０２を読

み込み、制御フローグラフ（ＣＦＧ：control flow graph）を作成する。

【００４７】

制御フローグラフは、プログラムの制御構造をグラフで表現したものであり、判定も分岐もせず順次実行する命令文の列から構成される基本ブロックノードと、基本ブロックノード間の制御の推移関係を示すエッジ等から構成される。制御フローグラフを生成する技術は、コンパイラ等に用いられ公知であるが、例えば、中央演算装置２は、監視対象プログラム１０２を記載した文字列を、プログラム言語として意味を持つ単位に分割し、当該分割した文字列の構成を、監視対象プログラム１０２が記載されたプログラム言語の文法に従って解析することで制御フローグラフを生成する。

【００４８】

なお、制御フローグラフの生成に関しては、例えば、“A.V.Aho, R.Sethi and J.D. Ullman. Compilers:Principles,Techniques,and Tools,Addisoin-Wesley,1986.”に記載されている。

【００４９】

図１１（ａ）に示す監視対象プログラム１１０１の制御フローグラフの一例を図５に示す。図５において、制御フローグラフは、基本ブロックノード５０１～５０３を備える。基本ブロックノード５０１は、関数「main」内の命令文「foo()」を備える。基本ブロックノード５０２は、関数「foo」内の命令文「gets()」及び「return」を備える。基本ブロックノード５０３は、命令文「exec」及び「exit」を備える。

【００５０】

中央演算装置２は、生成した制御フローグラフを変数Ｇに格納する。さらに、変数Ｇに格納した制御フローグラフを構成する基本ブロックノード集合を変数Ｎに格納する（図４におけるステップ４０１）。例えば、図５に一例を示す制御フローグラフを用いて説明すると、中央演算装置２は、変数Ｇに、図５に示す制御フローグラフを格納し、変数Ｎには、図５に示す制御フローグラフの基本ブロックノード集合、即ち、基本ブロックノード５０１、基本ブロックノード５０２、基本ブロックノード５０３を格納する。

【００５１】

次に、中央演算装置２は、基本ブロックノード集合Ｎが空集合（Ｎ＝Φ）であるか否か判定する（図４におけるステップ４０２）。

【００５２】

基本ブロックノード集合Ｎが空集合である（Ｎ＝Φ）場合、中央演算装置２は、後述する処理で取得するシステムコールと、当該システムコールまでの経路情報とからマッチング用文法１０７を生成する（図４におけるＳ４０８）。

【００５３】

基本ブロックノード集合Ｎが空集合でない（Ｎ≠Φ）場合、中央演算装置２は、基本ブロックノード集合Ｎから１つの基本ブロックノードを取り出し、変数ｎに格納する。さらに、基本ブロックノードｎに到達する分岐集合を変数Ｔに格納する。さらに、基本ブロックノードｎ中の分集合を変数Ｓに格納する（図４におけるステップ４０３）。

【００５４】

例えば、図５に一例を示す制御フローグラフを用いて説明すると、基本ブロックノード集合Ｎに基本ブロックノード５０１～５０３が格納されている場合、中央演算装置２は、Ｎ≠Φであるので、基本ブロックノード集合Ｎから基本ブロックノード５０１を取り出してｎとする。これにより、基本ブロックノード集合Ｎは、基本ブロックノード５０２及び５０３となる。ｎ（基本ブロックノード５０１）に到達する分岐集合は無いので、分岐集合Ｔ＝Φ（空集合）とする。ｎ中の命令文は「foo()」であるので、これを文集合Ｓとする。

【００５５】

次に、中央演算装置２は、基本ブロックノードｎ中の文集合Ｓが空集合（Ｓ＝Φ）であるか否か判定する（図４におけるステップ４０４）。

【００５６】

基本ブロックノードｎ中の文集合Ｓが空集合である（Ｓ＝Φ）場合、中央演算装置２は、ステップ４０２の処理を再度実行する。

【００５７】

基本ブロックノードｎ中の文集合Ｓが空集合でない（Ｓ≠Φ）場合、中央演算装置２は、基本ブロックノード中の文集合Ｓから１の文を取り出し、変数ｓに格納する（図４におけるステップ４０５）。

【００５８】

例えば、図５に一例を示す制御フローグラフを用いて説明すると、基本ブロックノードｎが基本ブロックノード５０１である場合、中央演算装置２は、Ｓ≠Φであるので、文集合Ｓから「foo()」を取り出して文ｓとする。文集合Ｓは、「foo()」を取り出したことにより空集合となる。

【００５９】

次に、中央演算装置２は、文ｓがシステムコールを実行する文であるか否か判定する（図４におけるステップ４０６）。

【００６０】

文ｓがシステムコールを実行する文でない場合、中央演算装置２は、ステップ４０４の処理を再度実行する。

【００６１】

文ｓがシステムコールを実行する文である場合、中央演算装置２は、文ｓに到達する分岐集合を「Ｔ+｛ｓ｝」とし（図４におけるステップ４０７）、当該情報をメモリ３内の特定の記憶部に格納した後、ステップ４０４の処理を再度実行する。なお、文ｓに到達する分岐集合「Ｔ+｛ｓ｝」の「Ｔ」は、基本ブロックノードｎに至るまでの分岐集合であり、「｛ｓ｝」は、システムコールである文ｓの分岐である。即ち、ここで得られた分岐集合「Ｔ+｛ｓ｝」が、文ｓのシステムコールまでの経路情報である。

【００６２】

例えば、図５に一例を示す制御フローグラフを用いて説明すると、文ｓが「foo()」である場合、中央演算装置２は、システムコールを実行する文ではないと判定し、ステップ４０４の処理を再度実行する。

【００６３】

以下、図５に一例を示す制御フローグラフを用いて、上述したステップによる処理を続けて説明する。中央演算装置２は、ステップ４０４の処理を再度実行する。文集合Ｓは、「foo()」を取り出したことにより空集合（Ｓ＝Φ）であるので、中央演算装置２は、ステップ４０２の処理を再度実行する。即ち、基本ブロックノード集合ＮがＮ≠Φであるので、基本ブロックノード集合Ｎから基本ブロックノード５０２を取り出して基本ブロックノードｎに格納する。これにより、基本ブロックノード集合Ｎは、基本ブロックノード５０３のみとなる。基本ブロックノードｎ（基本ブロックノード５０２）に到達する分岐集合は「main+m0」であるので、中央演算装置２は、これを分岐集合Ｔとする。さらに、ｎ中の命令文は「gets()」,「return」であるので、これを文集合Ｓとする（図４におけるステップ４０３）。

【００６４】

次に、中央演算装置２は、基本ブロックノード中の文集合Ｓが空集合（Ｓ＝Φ）であるか否か判定する（図４におけるステップ４０４）。この例では、文集合Ｓは空集合でない（Ｓ≠Φ）ので、文集合Ｓから命令文「gets()」を取り出して文ｓとする。文集合Ｓには、命令文「gets()」を取り出したことにより命令文「return」が残る。

【００６５】

次に、中央演算装置２は、文ｓがシステムコールを実行する文であるか否か判定する（図４におけるステップ４０６）。この例では、文ｓ（gets()）はシステムコールを実行する文であるので、文ｓに到達する分岐集合を「Ｔ+｛ｓ｝」、即ち「main+m0,foo+n0」とする（図４におけるステップ４０７）。中央演算装置２は、経路情報である分岐集合をメモリ３内の特定の記憶部に格納等した後、ステップ４０４の処理を再度実行する。

【００６６】
中央演算装置２は、上述した処理を繰り返し、監視対象プログラム１０２のすべてのシステムコールの経路情報を取得する。ステップ４０２の処理にて基本ブロックノード集合Ｎが空集合である（Ｎ＝Φ）場合、即ち、全てのシステムコールに関し経路情報を取得した場合、中央演算装置２は、上記した処理により取得したシステムコールと、当該システムコールまでの経路情報とからマッチング用文法１０７を生成する（図４におけるＳ４０８）。具体的には、例えば、中央演算装置２は、制御フローグラフＧの基本ブロックノードからシステムコールを順次抽出し、予め定めておいた書式に従う等して文法表現に変換する。さらに、中央演算装置２は、各システムコールまでの経路情報（分岐集合「Ｔ+｛ｓ｝」）を特定の記憶部から読み出して順次追加し、マッチング用文法１０７を生成する。

【００６７】
図６に、中央演算装置２が解析プログラム１０６を実行することにより、図１１（ａ）に示す監視対象プログラム１１０１から生成したマッチング用文法の一例を示す。図６において、マッチング用文法６０１は、「gets」というシステムコールを「main+m0」、「foo+n0」という順で分岐アドレスを経由して実行することを示している。「exec」というシステムコールを「main+m1」という分岐アドレスを経由して実行することを示している。「exit」というシステムコールを「main+m2」という分岐アドレスを経由して実行することを示している。

【００６８】
監視対象プログラム１０２を監視する動作を図７及び図８を参照して説明する。図７では監視プログラム１０１の動作を、図８では経路取得プログラム１０４を拡張モジュールとして実現するＯＳ１０３の動作を説明する。なお、図７を参照して説明する動作は、中央演算装置２が監視プログラム１０１をメモリ３にロードして実行することにより実現される動作である。また、図８を参照して説明する動作は、中央演算装置２が、経路取得プログラム１０４を拡張モジュールとするＯＳ１０３をメモリ３にロードして実行することにより実現する動作である。

【００６９】
ここでは、中央演算装置２は、ＯＳ１０３及び経路取得プログラム１０４により、少なくとも監視対象プログラム１０２を含む、侵入検知装置１で実行する全て又は一部のプログラムで分岐が発生する度に、当該分岐の分岐に関する情報を経路記憶部１０５に格納しておくものとする。

【００７０】
監視プログラム１０１の実行により、中央演算装置２は、マッチング用文法１０７をメモリ３から読み込んで、変数Ｇに格納する（図７における７０１）。この状態で、システムコールを示す情報と、当該システムコールまでの経路情報とを取得するまで待機する。監視プログラム１０１の実行開始は、入力部６を用いて利用者等が指示してもよく、また、タイマによる実行開始時間の設定や、侵入検知装置１の起動又はＯＳ１０３の起動と同時に実行されるようにしてもよい。

【００７１】
利用者の実行要求又はタイマによる開始時間の設定等により、中央演算装置２は、監視対象プログラム１０２を実行する。

【００７２】
監視対象プログラム１０２によりシステムコールを要求すると、中央演算装置２は、図８に一例を示す動作を、ＯＳ１０３を実行することで実現する。まず、中央演算装置２は、監視対象プログラム１０２がトレース中であるか否か判定する（図８におけるステップ８０１）。

【００７３】
この判定のために、例えば、中央演算装置２は、監視対象プログラム１０２を実行した場合、ＯＳ１０３の実行により、監視対象プログラム１０２を実行したことを通知するフ

ラグを立てる等する。中央演算装置２は、このフラグを確認する等して、監視対象プログラム１０２がトレース中であるか否か判定する。

【００７４】

監視対象プログラム１０２がトレース中でない場合、中央演算装置２は、ＯＳ１０３を実行することにより、要求したシステムコールを実行する（図８におけるステップ８０４）。

【００７５】

監視対象プログラム１０２がトレース中である場合、中央演算装置２は、ＯＳ１０３の実行により、経路記憶部１０５から、要求したシステムコールまでの経路情報を読み出し、要求したシステムコールを示す情報を変数Ｓに、読み出した経路情報を変数Ｔに格納する（図８におけるステップ８０２）。ここでは、中央演算装置２は、経路記憶部１０５内に格納している複数の分岐に関する情報から、要求したシステムコールまでの分岐アドレスを読み出し、当該情報を経路情報として変数Ｔに格納する。

【００７６】

このようにシステムコールを示す情報Ｓ及び経路情報Ｔを取得すると、中央演算装置２は、監視プログラム１０１の実行により、システムコールを示す情報を変数Ｓに、当該システムコールまでの経路情報を変数ＢＨに格納する（図７におけるステップ７０２）。ここでは、経路情報として、上述したシステムコールまでの分岐アドレスを示す情報がＢＨに格納される。

【００７７】

次に、中央演算装置２は、システムコールを示す情報Ｓが「Ｓ＝ｅｘｉｔ」であるか否か判定する（図７におけるステップ７０３）。

【００７８】

「Ｓ＝ｅｘｉｔ」である場合、監視対象プログラム１０２は終了するので、中央演算装置２は、監視プログラム１０１の処理を終了する。

【００７９】

「Ｓ＝ｅｘｉｔ」でない場合、中央演算装置２は、変数Ｇに格納したマッチング用文法Ｇを参照し、システムコールを示す情報Ｓと、マッチング用文法Ｇのシステムコールとが一致するか否か判定する（図７におけるステップ７０４）。

【００８０】

システムコールを示す情報Ｓとマッチング用文法Ｇのシステムコールとが一致しない場合、中央演算装置２は、監視対象プログラム１０２が不正なプログラムにより制御を奪われた可能性があると判定し、侵入警告の出力、又は、監視対象プログラム１０２の停止やスローダウンを行う（図７におけるステップ７０７）。侵入警告の出力とは、例えば、ディスプレイへの侵入警告の表示、スピーカへの警告音の出力、プリンタへの出力、通信ネットワークにより接続されたＰＣ（Personal Computer）や携帯電話等の通信装置（図示略）への電子メールや音声による通知等である。また、中央演算装置２は、侵入警告の出力、監視対象プログラム１０２の停止、監視対象プログラム１０２のスローダウンのうち全て又は一部を実行してもよい。

【００８１】

システムコールを示す情報Ｓとマッチング用文法Ｇのシステムコールとが一致する場合、中央演算装置２は、経路情報ＢＨと、マッチング用文法Ｇの経路情報とが一致するか否か判定する（図７におけるステップ７０５）。ここでは、中央演算装置２は、システムコールを示す情報Ｓまでの分岐アドレスである経路情報ＢＨと、ステップ７０４にて判定したマッチング用文法Ｇのシステムコールに対応する分岐アドレスとが一致するか否か判定する。

【００８２】

経路情報ＢＨとマッチング用文法Ｇの経路情報とが一致しない場合、中央演算装置２は、不正なプログラムにより監視対象プログラム１０２の制御が奪われた可能性があると判定し、侵入警告の出力、又は、監視対象プログラム１０２の停止又はスローダウンを行う

（図７におけるステップ７０７）。

【００８３】

経路情報ＢＨとマッチング用文法Ｇの経路情報とが一致する場合、中央演算装置２は、監視対象プログラム１０２が不正な侵入を受けていないと判定し、監視対象プログラム１０２の実行再開要求を行う（図７におけるステップ７０６）。そのために、中央演算装置２は、例えば、上述の判定結果を示すフラグを立てる等する。

【００８４】

ＯＳ１０３の実行により、中央演算装置２は、要求したシステムコール及び当該システムコールまでの経路情報が、マッチング用文法のシステムコール及び当該システムコールの経路情報と一致するか否かの判定結果により、システムコールが実行可能か否か判定する（図８におけるステップ８０３）。判定結果は、例えば、監視プログラム１０１の実行により立てられた判定結果を示すフラグ、又は、侵入警告の出力を実行するためのフラグや信号等の有無、又は、監視対象プログラム１０２の停止やスローダウン等を実行するためのフラグや信号の有無等を確認することにより取得することができる。

【００８５】

判定結果が、監視対象プログラム１０２が不正な侵入を受けていないと判定したことを示すものである場合、中央演算装置２は、要求したシステムコールを実行する（図８におけるステップ８０４）。

【００８６】

判定結果が、不正なプログラムにより監視対象プログラム１０２の制御が奪われた可能性があると判定されたことを示すものである場合、中央演算装置２は、図８に一例を示すＯＳ１０３の実行を終了する。

【００８７】

中央演算装置２は、監視対象プログラム１０２の実行により別のシステムコールを要求する場合でも、上述した処理を再度実行することにより、システムコールと当該システムコールまでの経路情報とのマッチングを行う。その場合、中央演算装置２は、例えば、次に実行するはずのシステムコール及び当該システムコールに対応する経路情報をマッチング用文法１０７から読み出してメモリ３等の特定の記憶部に格納し、要求したシステムコールを示す情報及び当該システムコールまでの経路情報を、当該特定の記憶部に格納したシステムコール及び当該システムコールまでの経路情報と比較することにより、マッチングを行ってもよい。

【００８８】

上述のように動作することで、侵入検知装置１への不正侵入を検知することが可能となる。図９に、中央演算装置２がＯＳ１０３を実行することにより取得する、システムコール及び当該システムコールまでの経路情報の一例を示す。図９において、情報９０１は、図１１（ａ）に一例を示す監視対象プログラム１１０１が不正プログラムにより攻撃されていない場合に、中央演算装置２が取得する情報の一例である。また、図９の情報９０２は、図１１（ａ）に一例を示す監視対象プログラム１１０１が不正プログラムにより攻撃された場合に、中央演算装置２が取得する情報の一例である。なお、この不正プログラムにより攻撃された場合とは、図１１（ｄ）に示すような、「gets」の呼び出しで、配列「buf」１１０３の領域長１０バイトを超えた不正なデータが読み込まれ、関数「foo」の戻りアドレスが、攻撃コード１１０５のアドレスを示すように変更される場合を示す。

【００８９】

中央演算装置２が図９（ａ）に一例を示す情報９０１を取得する場合、図６に一例を示すマッチング用文法６０１のシステムコール及び経路情報と、図９（ａ）に一例を示す情報９０１とのシステムコール及び経路情報とは一致する。従って、中央演算装置２は、監視対象プログラム１０２が不正なプログラムにより攻撃されていないと判定する。

【００９０】

中央演算装置２が図９（ｂ）に一例を示す情報９０２を取得する場合、図６に一例を示すマッチング用文法６０１のシステムコール及び経路情報と、図９（ｂ）に一例を示す情

報９０２とのシステムコール及び経路情報とは一致しない。即ち、「exec」というシステムコールまでの経路情報が、図６に一例を示すマッチング用文法６０１では「main+m1」であるのに対し、図９（ｂ）に一例を示す情報９０２では「STACK+x0」である。また、「exit」というシステムコールまでの経路情報が、図６に一例を示すマッチング用文法６０１では「main+m2」であるのに対し、図９（ｂ）に一例を示す情報９０２では「STACK+x2」である。このようにシステムコールまでの経路情報が異なるので、中央演算装置２は、監視対象プログラム１０２が不正なプログラムにより攻撃された可能性があると判定する。その場合、中央演算装置２は、侵入警告の出力や、監視対象プログラム１０２の停止やスローダウンを実行する。

【００９１】

このように、システムコールが、元のプログラムと、不正プログラムにより攻撃された後のプログラムとで一致する場合でも、不正侵入を検知することが可能となる。

【００９２】

以上、この発明の実施形態を図面を参照して詳述してきたが、具体的な構成はこの実施形態に限られるものではなく、この発明の要旨を逸脱しない範囲の設計変更等も含まれる。

【００９３】

例えば、上述の実施形態では、経路情報は、分岐トレースバッファの機能により取得する分岐アドレスであるものとしたが、これに限られるものではない。経路情報の例として、プログラムを、特定の大きさに区切ったページのページ番号、メモリ領域の種別（コード領域やスタック領域等）などを用いてもよい。

【００９４】

また、経路情報は分岐トレースバッファの機能により取得するものとしたが、これに限られたわけではない。例えば、監視対象プログラムに、分岐の経路情報を取得してメモリ等に格納する機能を備えておき、分岐命令を実行する前に当該機能を実行することにより、経路取得プログラムの機能を実現させてもよい。その場合、経路情報を取得するのはＯＳの実行に限られるものではなく、例えば、監視プログラムを実行することにより経路情報を取得してもよい。

【００９５】

また、上述の実施形態では、監視対象プログラムの実行をマッチング用文法と比較して監視するものとしたが、必ずしも文法表現であるものと比較する必要はない。監視対象プログラムが不正プログラムにより攻撃されていない場合のシステムコールと、当該システムコールまでの経路情報とが対応付けられていればよい。例えば、システムコール及び経路情報と、当該システムコールによる制御の遷移状態を示す状態遷移表等を参照して、監視対象プログラムを監視してもよい。

【図面の簡単な説明】

【００９６】

【図１】本実施形態の侵入検知装置の構成の一例を示す図である。
【図２】同実施形態において、分岐に関する情報の例を示す図である。
【図３】同実施形態において、マッチング用文法の一例を示す図である。
【図４】同実施形態において、マッチング用文法を生成する動作例を説明する図である。
【図５】同実施形態において、制御フローグラフの一例を説明する図である。
【図６】同実施形態において、マッチング用文法の一例を示す図である。
【図７】同実施形態において、監視プログラムの動作例を説明する図である。
【図８】同実施形態において、ＯＳの動作例を説明する図である。
【図９】同実施形態において、監視対象プログラムが実行された場合のシステムコールと、当該システムコールの経路情報との例である。
【図１０】従来の侵入検知の方法を説明する図である。
【図１１】従来の侵入検知方法において、不正侵入を検知できない例を説明する図である。

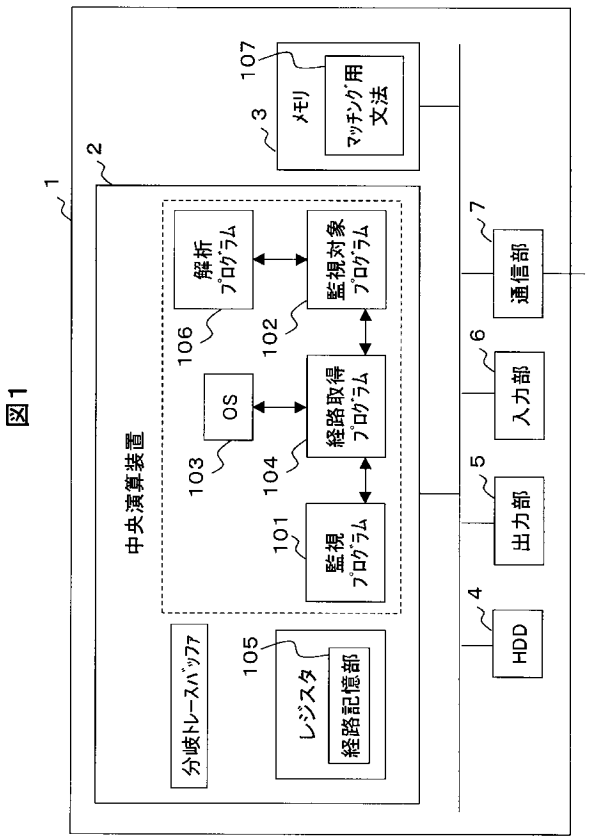
【符号の説明】

【００９７】

１： 侵入検知装置、２：中央演算装置、３：メモリ、４：ＨＤＤ、５：出力部、６：入力部、７：通信部、１０１：監視プログラム、１０２：監視対象プログラム、１０３：ＯＳ、１０４：経路取得プログラム、１０５：経路記憶部、１０６：解析プログラム、１０７：マッチング用文法、２０１：分岐に関する情報、３０１：マッチング用文法、５０１：基本ブロックノード、５０２：基本ブロックノード、５０３：基本ブロックノード、６０１：マッチング用文法、９０１：システムコールと当該システムコールまでの経路情報、９０１：システムコールと当該システムコールまでの経路情報、１００１：監視対象プログラム、１００２：マッチング用文法、１００３：監視プログラム、１１０１：監視対象プログラム、１１０２：マッチング用文法、１１０３：配列「buf」、１１０４：配列「PC」、１１０５：攻撃コード

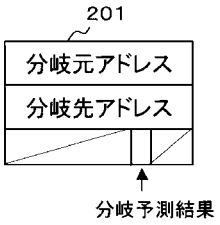
【図１】

図1



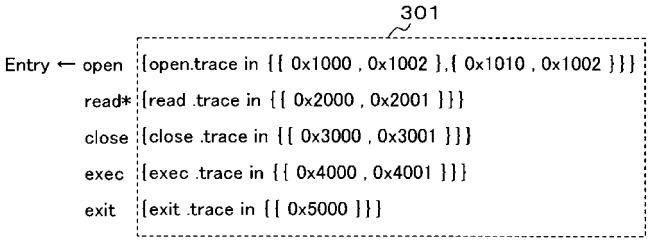
【図２】

図2



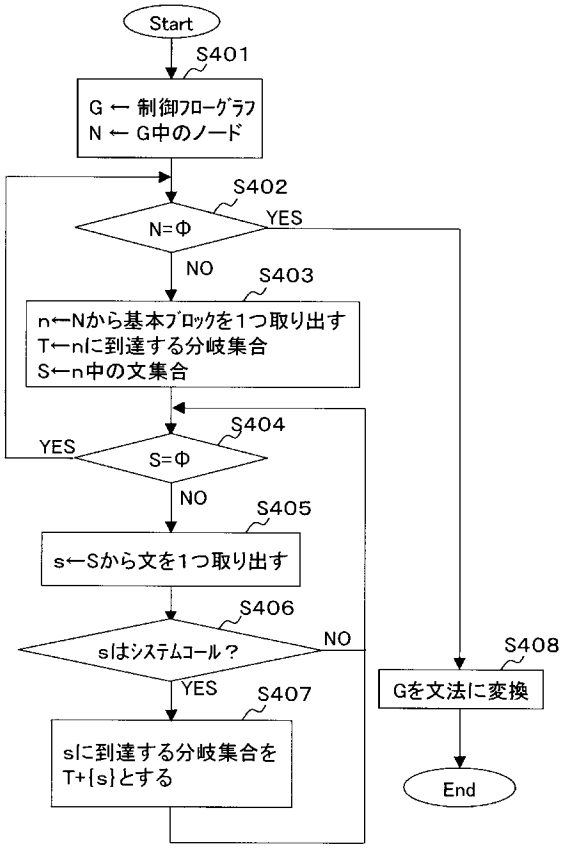
【図３】

図3



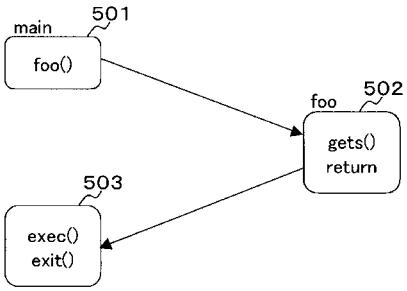
【図４】

図4



【図５】

図5



【図６】

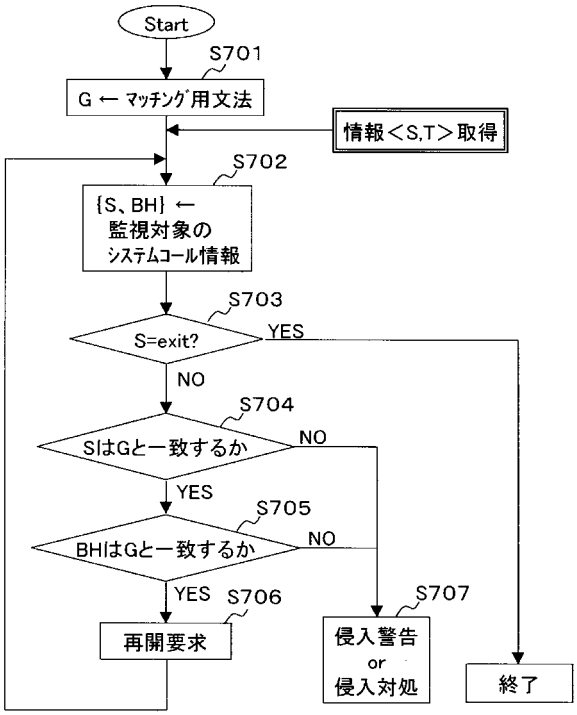
図6

601

```
Entry ← gets { assert ( gets.trace in { main+m0 , foo+n0 } ) }
        exec{ assert ( exec .trace in { main+m1 } ) }
        exit { assert ( exit .trace in { main+m2 } ) }
```

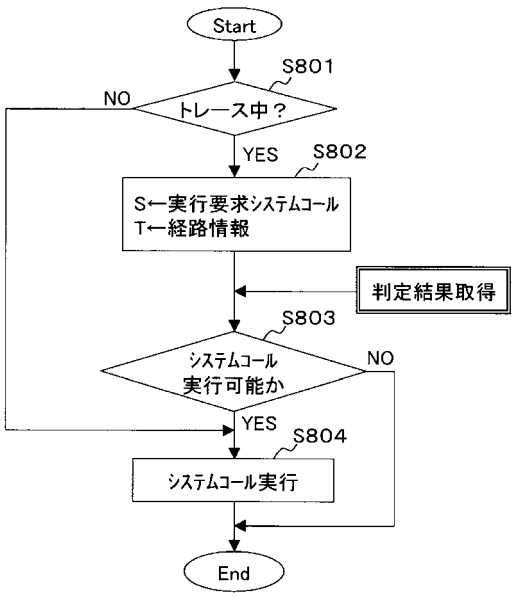
【図７】

図7



【図８】

図8

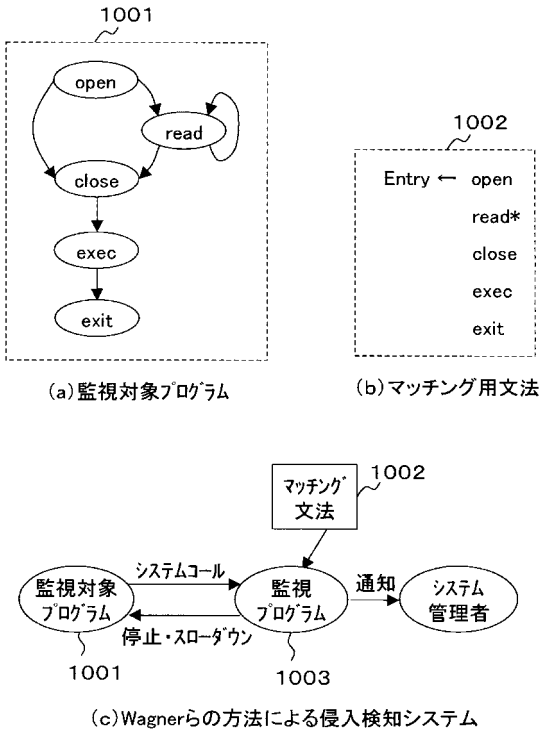


【図９】

図9

【図１０】

図10



【図１１】

図11