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

特許第5411966号
(P5411966)

(45)発行日　平成26年2月12日(2014.2.12)　(24)登録日　平成25年11月15日(2013.11.15)

(51)Int.Cl.		F I		
G06F 21/53	(2013.01)	G06F	21/00	153
G06F 9/46	(2006.01)	G06F	9/46	350
G06F 9/54	(2006.01)	G06F	9/46	480Z

請求項の数 5　(全 19 頁)

(21)出願番号　特願2012-154668(P2012-154668)
(22)出願日　平成24年7月10日(2012.7.10)
(65)公開番号　特開2014-16877(P2014-16877A)
(43)公開日　平成26年1月30日(2014.1.30)
審査請求日　平成25年2月22日(2013.2.22)

(73)特許権者　000004226
日本電信電話株式会社
東京都千代田区大手町二丁目３番１号
(74)代理人　100089118
弁理士　酒井　宏明
(74)代理人　100112656
弁理士　宮田　英毅
(72)発明者　秋山　満昭
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内
(72)発明者　川古谷　裕平
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内

審査官　石田　信行

最終頁に続く

(54)【発明の名称】監視装置および監視方法

(57)【特許請求の範囲】
【請求項１】
監視対象のプロセスからコンピュータの資源へのアクセス要求が検知された場合に、当該アクセス要求の種類に基づいて、当該アクセス要求が前記コンピュータの資源に影響を与えるものであるか否かを判定するアクセス監視部と、
前記アクセス監視部によって前記アクセス要求が前記資源に影響を与えるものでないと判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、要求された資源に当該プロセスをアクセスさせ、一方、前記資源に影響を与えるものであると判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、当該アクセス要求によって要求された資源と同一の仮想資源を当該アクセス要求の要求元のプロセスごとに作成し、要求された資源の代わりに、作成した仮想資源に当該プロセスをアクセスさせるアクセス制御部と、
プロセスの生成要求が検知された場合に、当該プロセスの生成要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定する

プロセス監視部と、
前記プロセス監視部によって関連プロセスであると判定された場合には、前記既に生成されたプロセスに対応する仮想資源のアドレスを前記関連プロセスに通知し、該仮想資源を前記関連プロセスが参照するように制御するプロセス制御部と、
を備えたことを特徴とする監視装置。
【請求項２】
前記プロセス監視部は、前記プロセスの生成要求に基づいて生成されたプロセスが、既に生成されたプロセスの子プロセスであるか否かを判定し、
前記プロセス制御部は、前記プロセス監視部によって子プロセスであると判定された場合には、前記既に生成されたプロセスに対応する仮想資源のアドレスを前記子プロセスに通知し、該仮想資源を前記子プロセスが参照するように制御することを特徴とする請求項１に記載の監視装置。
【請求項３】
前記アクセス要求が前記仮想資源を削除するものであった場合には、該仮想資源に対応する削除フラグを設定する仮想資源管理部をさらに備え、
前記プロセス制御部は、前記仮想資源管理部によって削除フラグが設定された仮想資源へのアクセスがあった場合には、該アクセスを拒否することを特徴とする請求項１または２に記載の監視装置。
【請求項４】
前記アクセス要求が前記仮想資源をコピーするものであった場合には、該仮想資源のコピーを行う代わりにコピー先の仮想資源として、既に作成された仮想資源を対応付けて管理する仮想資源管理部をさらに備えることを特徴とする請求項１または２に記載の監視装置。
【請求項５】
監視装置で実行される監視プログラムであって、
監視対象のプロセスからコンピュータの資源へのアクセス要求が検知された場合に、当該アクセス要求の種類に基づいて、当該アクセス要求が前記コンピュータの資源に影響を与えるものであるか否かを判定するアクセス監視工程と、
前記アクセス監視工程によって前記アクセス要求が前記資源に影響を与えるものでないと判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、要求された資源に当該プロセスをアクセスさせ、一方、前記資源に影響を与えるものであると判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、当該アクセス要求によって要求された資源と同一の仮想資源を当該アクセス要求の要求元のプロセスごとに作成し、要求された資源の代わりに、作成した仮想資源に当該プロセスをアクセスさせるアクセス制御工程と、
プロセスの生成要求が検知された場合に、当該プロセスの生成要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定するプロセス監視工程と、
前記プロセス監視工程によって関連プロセスであると判定された場合には、前記既に生成されたプロセスに対応する仮想資源のアドレスを前記関連プロセスに通知し、該仮想資源を前記関連プロセスが参照するように制御するプロセス制御工程と、
を含んだことを特徴とする監視方法。
【発明の詳細な説明】
【技術分野】

【０００１】
本発明は、監視装置および監視方法に関する。
【背景技術】
【０００２】
従来、コンピュータウィルスなどの悪性プログラムや、脆弱性を有する可能性のある信頼できないプログラムを仮想マシン上に構築したＯＳ（Operating System）環境で実行し、それらの振る舞いを監視する手法が知られている（例えば、非特許文献１、非特許文献２参照）。仮想マシン上のＯＳ環境で不正プログラムを実行することによって、不正プログラムがＯＳを破壊する振る舞いをしたとしても、その影響が仮想マシンを動作させているホストマシンまで及ばないようにすることができる。すなわち、ホストマシンから隔離された環境で、不正プログラムを実行することができる。
【０００３】
一方、プロセス単位で、信頼できないプロセスとその実行環境であるホストマシンを隔離する手法も存在している。この手法では、所定のサブディレクトリを仮想的なルートディレクトリとして監視対象プロセスに見せかけ、そのルートディレクトリ構造以外のファイルへのアクセスは一切禁止することにより、プロセスと、その実行環境を提供しているホストマシンとの隔離を実現している（例えば、非特許文献３参照）。
【０００４】
しかし、仮想マシン上にＯＳ環境を構築して実行環境を作成する手法は、隔離といった点では有用ではあるが、プログラム一つを解析するためにＯＳ環境一つを用意する必要があり、コンピュータ資源に要する多大なコストがかかる。また、プロセス単位で隔離する方法では、完全に隔離された不自然なコンピュータ環境上でプログラムが実行されるため、隔離された環境で実行されていることが不正プログラムによって容易に検知されてしまい、それによる不正プログラムが動作を停止するなどの解析が阻害される可能性がある。また、隔離された環境で不正プログラムを正常に動作させるには、その不正プログラムが利用する仮想資源（ファイルやレジストリなど）と同一の環境を用意する必要があり、これも非常にコストがかかる。そのため、大量のプログラムを同一コンピュータ環境上で監視するには適していない。
【０００５】
そこで、プロセス単位あるいはプログラムを構成するモジュール単位でアクセス要求があった資源に対して、システムに影響をあたえるアクセスであると判断された場合に、当該資源と同一になるように作成した仮想資源を生成する技術が知られている（特許文献１参照）。これによって、プロセスやモジュールを仮想的に隔離環境で実行することができ、また隔離環境であることが不正プログラムに検知されることなく、システムの破壊しようとする動作であっても仮想資源の作成および変更にとどめて実際の実行環境に影響を与えない仕組みである。
【先行技術文献】
【特許文献】
【０００６】
【特許文献１】特開２０１０－０２０７１３号公報
【非特許文献】
【０００７】
【非特許文献１】Paul Barham、Boris Dragovic、Keir Fraser、Steven Hand、Tim Harris、Alex Ho、Rolf Neugebauer、Ian Pratt、Andrew Warfield、”Xen and the Art of Virtualization”、University of Cambridge Computer Laboratory、［online］、［平成２４年６月２９日検索］、インターネット＜URL:http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf＞
【非特許文献２】”ＶＭｗａｒｅ ｗｏｒｋｓｔａｔｉｏｎ”、［online］、ヴイエムウェア株式会社、［平成２４年６月２９日検索］、インターネット＜URL:http://www.vmware.com/jp/virtualization/＞

【非特許文献３】ＦｒｅｅＢＳＤ、「ＪＡＩＬ」、［online］、［平成２４年６月２９日検索］、インターネット＜URL:http://www.freebsd.org/cgi/man.cgi?query=jail&format=html＞

【発明の概要】

【発明が解決しようとする課題】

【０００８】

しかしながら、従来の技術では、プロセスを仮想的な隔離環境で実行した場合において、異なるプロセス間で資源（ファイルやレジストリ等）を共有することができないという課題があった。このため、異なるプログラムやモジュール間で共通の資源を用いて連携した処理を行いたい場合に、共通の資源を参照できないため本来プログラムが意図している動作が実行できなかった。例えば、ＷｅｂブラウザがＰＤＦリーダーと連携してＰＤＦファイルをレンダリングする場合である。ＷｅｂブラウザがインターネットからダウンロードしたＰＤＦファイルをＰＤＦリーダーで自動的に読み込む際に、Ｗｅｂブラウザのプロセスが作成したＰＤＦファイルの仮想資源をＰＤＦリーダーのプロセスが参照できないため、ＰＤＦファイルのレンダリングに失敗してしまう。

【０００９】

そこで、この発明は、上述した従来技術の課題を解決するためになされたものであり、プロセスを仮想的な隔離環境で実行した場合においても、異なるプロセス間で資源を共有することができるようにすることを目的とする。

【課題を解決するための手段】

【００１０】

上述した課題を解決し、目的を達成するため、監視装置は、監視対象のプロセスからコンピュータの資源へのアクセス要求が検知された場合に、当該アクセス要求の種類に基づいて、当該アクセス要求が前記コンピュータの資源に影響を与えるものであるか否かを判定するアクセス監視部と、前記アクセス監視部によって前記アクセス要求が前記資源に影響を与えるものでないと判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、要求された資源に当該プロセスをアクセスさせ、一方、前記資源に影響を与えるものであると判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、当該アクセス要求によって要求された資源と同一の仮想資源を当該アクセス要求の要求元のプロセスごとに作成し、要求された資源の代わりに、作成した仮想資源に当該プロセスをアクセスさせるアクセス制御部と、プロセスの生成要求が検知された場合に、当該プロセスの生成要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定するプロセス監視部と、前記プロセス監視部によって関連プロセスであると判定された場合には、前記既に生成されたプロセスに対応する仮想資源のアドレスを前記関連プロセスに通知し、該仮想資源を前記関連プロセスが参照するように制御するプロセス制御部と、を備えたことを特徴とする。

【００１１】

また、監視方法は、監視対象のプロセスからコンピュータの資源へのアクセス要求が検知された場合に、当該アクセス要求の種類に基づいて、当該アクセス要求が前記コンピュータの資源に影響を与えるものであるか否かを判定するアクセス監視工程と、前記アクセス監視工程によって前記アクセス要求が前記資源に影響を与えるものでないと判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成さ

れていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、要求された資源に当該プロセスをアクセスさせ、一方、前記資源に影響を与えるものであると判定された場合には、当該アクセス要求によって要求された資源に対応し、かつ、当該アクセス要求の要求元のプロセスに対応する仮想資源がすでに作成されているか否かを判定し、作成されていたときには、要求された資源の代わりに、当該アクセス要求の要求元のプロセスに対応する仮想資源に当該プロセスをアクセスさせ、作成されていないときには、当該アクセス要求によって要求された資源と同一の仮想資源を当該アクセス要求の要求元のプロセスごとに作成し、要求された資源の代わりに、作成した仮想資源に当該プロセスをアクセスさせるアクセス制御工程と、プロセスの生成要求が検知された場合に、当該プロセスの生成要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定するプロセス監視工程と、前記プロセス監視工程によって関連プロセスであると判定された場合には、前記既に生成されたプロセスに対応する仮想資源のアドレスを前記関連プロセスに通知し、該仮想資源を前記関連プロセスが参照するように制御するプロセス制御工程と、を含んだことを特徴とする。

【発明の効果】

【００１２】

本願に開示する監視装置および監視方法は、プロセスを仮想的な隔離環境で実行した場合においても、異なるプロセス間で資源を共有することができるという効果を奏する。

【図面の簡単な説明】

【００１３】

【図１】図１は、本実施例に係る監視システムの構成を示すブロック図である。

【図２】図２は、監視部の構成を示す機能ブロック図である。

【図３】図３は、プロセス間で変換テーブルを共有する処理を説明する図である。

【図４】図４は、監視の処理負荷が軽減されることを説明する図である。

【図５】図５は、ファイル・レジストリの状態を管理する処理を説明する図である。

【図６】図６は、同一のファイルは同一のファイル実体として管理することを説明する図である。

【図７】図７は、プログラム実行部によって行われる処理の処理手順を示すフローチャートである。

【図８】図８は、仮想資源のアクセス処理の処理手順を示すフローチャートである。

【図９】図９は、プロセス制御部によって行われる処理の処理手順を示すフローチャートである。

【図１０】図１０は、監視プログラムを実行するコンピュータを示す図である。

【発明を実施するための形態】

【００１４】

以下に添付図面を参照して、この発明に係る監視装置および監視方法の好適な実施例を詳細に説明する。なお、この実施例によりこの発明が限定されるものではない。

【実施例】

【００１５】

まず、図１を用いて、本実施例に係る監視システムの概要について説明する。図１は、本実施例に係る監視システムの構成を示すブロック図である。同図に示すように、本実施例に係る監視システムは、ホストシステム１０と、監視部２０と、監視対象プログラム３０とを備える。なお、ここで説明する構成はあくまで一例であり、監視システムは他のさまざまな形態でも実施が可能である。

【００１６】

ホストシステム１０は、各種プログラムを動作させるためのＯＳや各種の資源が実装されたコンピュータである。ここで、「資源」とは、ホストシステム１０上でプログラムをＯＳ上で動作させるために必要な各種のリソースであり、たとえば、ファイルやレジストリ、カーネルデータ構造体などである。なお、以下では、ホストシステム１０に実装され

ている資源を「ホスト資源」と呼ぶ。

【００１７】

監視部２０は、ホストシステム１０上で動作するプログラムであり、監視対象プログラム３０からホスト資源やプロセスに関するアクセスを監視する。たとえば、この監視部２０は、複数のモジュールから構成されるモジュールライブラリとして実装される。各監視部２０は、プロセス単位（プログラム単位）またはモジュール単位で、監視対象プログラム３０の動作を監視する。

【００１８】

監視対象プログラム３０は、監視システムによる監視の対象となるプログラムであり、監視部２０と一緒に動作する。たとえば、この監視対象プログラム３０は、コンピュータウィルスやワームなどの悪性プログラムや、脆弱性を有する可能性がある信頼できないプログラムなどである。

【００１９】

このような構成のもと、本実施例では、監視部２０が、監視対象のプログラムを監視し、ホスト資源へのアクセス要求やプロセス要求を検知する。そして、監視部２０は、アクセス要求を検知した場合は、アクセス要求の種類に応じて、そのアクセス要求がホスト資源に影響を与えるものであるか否かを判定する。ここでホストに影響を与えるものとは、ホストシステム１０を破壊する可能性があるアクセス要求やＯＳの動作が変更される可能性があるアクセス要求である。ホストに影響を与えないものとしては、すでに存在する資源に対して読み取りのみを行うようなアクセス要求などである。

【００２０】

そして、監視部２０は、検知したアクセス要求がホスト資源に影響を与えるものでないと判断した場合は、そのアクセス要求によって要求されたホスト資源に監視対象のプログラムをアクセスさせる。一方、監視部２０は、検知したアクセス要求がホスト資源に影響を与えるものであると判断した場合は、そのアクセス要求によって要求されたホスト資源の代わりに、当該ホスト資源と同一になるように作成した「仮想資源」に監視対象プログラム３０をアクセスさせる。このとき、監視部２０は、アクセス要求がホスト資源を新規に作成するものであった場合には、そのホスト資源に対応する仮想資源を新規に作成する。

【００２１】

このように、本実施例では、監視部２０が、監視対象プログラム３０からホスト資源へのアクセス要求を検知した場合に、そのアクセス要求の種類に応じて、要求されたホスト資源またはそのホスト資源と同一になるように作成した仮想資源のいずれかに監視対象プログラム３０をアクセスさせる。

【００２２】

かかるアクセス要求の制御は、監視対象プログラム３０の外部で透過的に行われるため、監視対象プログラム３０からみると、あたかもホスト資源に正常にアクセスしているように見せかけることができるため、監視および制御によって監視対象のアクセス要求が妨げられることはない。

【００２３】

本実施例によれば、仮想ＯＳを用意する必要なく、監視対象のプログラムを他のプログラムとは仮想的に隔離しつつ実行できる。また、監視対象のプログラムに対する仮想資源は動作時に動的に作成するため、あらかじめホスト資源を用意する必要がない。これによって、同一ホストシステム上で大量の監視対象プログラムを同時に動作させることができるため、コストを削減できる。

【００２４】

そして、本実施例では、関係のあるプロセスには、同じ仮想資源を割り当て、関連するプロセス（例えば、Ｗｅｂブラウザとヘルパーアプリのプロセス）であれば、同一の仮想資源を参照できることを特徴の一つとする。

【００２５】

つまり、監視部２０は、監視対象プログラム３０からプロセス生成の要求が検知された場合に、当該プロセス生成の要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定する。

【００２６】

そして、監視部２０は、関連するプロセスであると判定された場合には、既に生成されたプロセスに対応する仮想資源のアドレスを関連プロセスに通知し、該仮想資源を関連プロセスが参照するように制御する。例えば、監視部２０は、親プロセスであるプロセスＡのプロセス生成イベントを監視し、プロセス生成イベントに基づいて生成された子プロセスＢに対して親プロセスであるプロセスＡと同様の変換テーブルの場所（例えば、共有メモリのアドレス）を通知し共有する。これにより、関連するプロセスであるプロセスＡとプロセスＢ間で共通のファイルやレジストリが参照できる。

【００２７】

次に、図２を用いて、監視部２０の構成について説明する。図２は、監視部２０の構成を示す機能ブロック図である。図２に示すように、監視部２０は、特に、プログラム実行部２１と、資源アクセス監視部２２と、資源アクセス制御部２３と、動作ログ保存部２４と、仮想資源処理部２５と、プロセス監視部２６と、プロセス制御部２７と、仮想資源管理部２８とを有する。なお、ここでは、各機能部が有する機能の概要について説明し、各機能部によって行われる処理については、後に詳細に説明する。

【００２８】

プログラム実行部２１は、監視部２０によって提供される半透過的な仮想隔離環境で監視対象プログラム３０を動作させるためのモジュールである。このプログラム実行部２１は、後述する監視用モジュールを監視対象プログラム３０に挿入することによって、監視システムが提供する半透過的な仮想隔離環境で監視対象プログラム３０を実行させる。

【００２９】

具体的には、プログラム実行部２１は、メモリ空間において、監視対象プログラム３０のコードに監視用モジュールのコードを挿入し、そのうえで、監視対象プログラム３０を実行する。または、監視モジュールをＤＬＬ（Dynamic Link Library）の一部としてホストシステム１０側に実装させておき、監視対象プログラム３０には、各監視モジュールを呼び出すためのＡＰＩ（Application Programming Interface）が記述されたコードを監視対象プログラム３０に挿入し、これにより、監視対象プログラム３０が実行時に監視モジュールを呼び出すようにしてもよい。

【００３０】

資源アクセス監視部２２は、監視対象プログラム３０からホスト資源へのアクセス要求を検知する監視用モジュールである。なお、資源アクセス監視部２２は、操作者によって、監視対象プログラムを構成するモジュールの中から監視対象のモジュールが指定されていた場合には、モジュール単位でアクセス要求を検知する。

【００３１】

資源アクセス制御部２３は、資源アクセス監視部２２によってアクセス要求が検知された場合に、当該アクセス要求の種類に基づいて、当該アクセス要求がホスト資源に影響を与えるものであるか否かを判定する監視用モジュールである。

【００３２】

動作ログ保存部２４は、仮想資源処理部２５によるアクセス制御の履歴を動作ログとして保存する記憶部である。具体的には、この動作ログ保存部２４は、監視対象プログラム３０からアクセス要求によって要求されたホスト資源と、当該ホスト資源に対応するものとして作成された仮想資源とを対応付けた情報を変換テーブルとして保存する。

【００３３】

仮想資源処理部２５は、資源アクセス制御部２３による判定の結果に基づいて、監視対象プログラム３０からホストシステム１０へのアクセスを制御する監視用モジュールである。

【００３４】

具体的には、仮想資源処理部２５は、資源アクセス制御部２３によって、監視対象プログラム３０からのアクセス要求がホスト資源に影響を与えるものでないと判定された場合には、当該アクセス要求によって要求されたホスト資源に監視対象プログラム３０をアクセスさせる。

【００３５】

一方、仮想資源処理部２５は、資源アクセス制御部２３によって、監視対象プログラム３０からのアクセス要求がホスト資源に影響を与えるものであると判定された場合には、当該アクセス要求によって要求されたホスト資源の代わりに、当該ホスト資源と同一になるように作成した仮想資源に監視対象プログラム３０をアクセスさせる。

【００３６】

そして、仮想資源処理部２５は、上記したアクセス制御を行った場合には、監視対象プログラム３０からアクセス要求によって要求されたホスト資源と、実際に監視対象プログラム３０をアクセスさせた仮想資源とを対応付けた情報を変換テーブルとして動作ログ保存部２４に保存する。

【００３７】

なお、仮想資源処理部２５は、操作者によって、監視対象プログラム３０を構成するモジュール３１の中から監視対象のモジュールが指定されていた場合には、指定されたモジュールについて、アクセス要求によって要求されたホスト資源または仮想資源へのアクセスを制御する。

【００３８】

プロセス監視部２６は、監視対象プログラム３０からプロセス生成の要求が検知された場合に、当該プロセス生成の要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定する。例えば、プロセス監視部２６は、プロセス生成の要求に基づいて生成されたプロセスが、既に生成されたプロセスの子プロセスであるか否かを判定する。なお、事前に関連するプロセス同士を予め定義しておき、予め定義された情報を用いて、生成されたプロセスが、既に生成されたプロセスと関連のあるプロセスであるか判定するようにしてもよい。

【００３９】

プロセス制御部２７は、プロセス監視部２６により関連するプロセスであると判定された場合には、既に生成されたプロセスに対応する仮想資源のアドレスを関連プロセスに通知し、該仮想資源を関連プロセスが参照するように制御する。例えば、プロセス監視部２７は、プロセス監視部２６によって子プロセスであると判定された場合には、既に生成されたプロセスに対応する仮想資源のアドレスを子プロセスに通知し、該仮想資源を子プロセスが参照するように制御する。

【００４０】

ここで、図３の例を用いて、プロセス監視部２６およびプロセス制御部２７の処理を説明する。図３は、プロセス間で変換テーブルを共有する例を示す図である。例えば、プロセス監視部２６が、親プロセスであるプロセスＡのプロセス生成イベントを監視する。そして、プロセス制御部２７は、プロセスＡのプロセス生成イベントに基づいて生成された子プロセスＢに対して親プロセスであるプロセスＡと同様の変換テーブル（共有メモリのアドレス）の場所を通知し共有させるように制御する。

【００４１】

また、同様に、プロセス制御部２７は、プロセスＢのプロセス生成イベントがあった場合に、プロセス生成イベントに基づいて生成された子プロセスＣに対して親プロセスであるプロセスＢと同様の変換テーブル（共有メモリのアドレス）の場所を通知し共有させるように制御する。これにより、関連するプロセスであるプロセスＡとプロセスＢとプロセスＣとの間で共通のファイルやレジストリが参照できる。

【００４２】

つまり、異なるプログラムやモジュールが共通の資源を参照して動作をするものであった場合、各プログラムやモジュール毎に作成された仮想資源のみを参照することによる資

源の見え方の違い（つまり、例えばプログラムＡからは仮想資源１が参照できるが、プログラムＢからは仮想資源１は参照できない）から二者間で不整合が発生するが、本実施例によれば、プログラムやモジュールの関連性を把握したうえで関連性があると判断できた場合は共通の仮想資源を参照させるため、協調動作を行う際の異なるプログラムもしくはモジュール間における資源の不整合を解消することができる。

【００４３】

また、プログラムやモジュールに対して関連性（例えば事前にリストで定義されたプログラムおよびモジュール、もしくは親プロセスと子プロセス）に基づいて資源の共有を自動的に行うことが可能である。

【００４４】

また、本実施例により、監視処理の処理負荷を軽減することができる。すなわち、図４に示すように、プロセスレベルでモニターを実装すると、監視対象のプロセスだけを検査すればよく監視処理の処理負荷を軽減することができる。一方、カーネルレベルでモニターを実装すると、監視対象外のプロセスのイベントまで検査しなければならない。また、カーネルレベルでモニターを実装すると、ある特定のプロセスＩＤのイベントだけを処理するにしても、すべてのイベントをトラップしてプロセスＩＤを識別する必要があり処理負荷が重い。

【００４５】

つまり、カーネルレイヤでの挙動監視を行う場合は、ＯＳ上のあらゆるアプリケーションやモジュールの挙動イベントを捕捉し監視対象のプログラムやモジュールを識別した上で処理を行うため、監視対象外のアプリケーションやモジュールの挙動イベントがオーバーヘッドとなる。一方、本実施例では、監視対象のプログラムやモジュールで挙動を観測するため、監視対象外のプログラムやモジュールの挙動イベントを捕捉せず効率的に監視が可能になる。

【００４６】

このように、特定のプロセスやモジュールについて仮想資源を共有することで、複数のプログラムやモジュールが連携した処理を実行する。これにより、資源と仮想資源の対応関係の情報を共有することで仮想資源共有を行う。ここで、資源と仮想資源の対応関係とは、”アクセス要求があった資源”と”アクセス要求があった資源と同一のものとして作成された仮想資源”の関係である。本実施例では、各プロセス毎に、仮想資源の対応関係の情報を「変換テーブル」で管理する。以下では、変換テーブルを管理する仮想資源管理部２８について説明する。

【００４７】

仮想資源管理部２８は、ファイルやレジストリの変換テーブルの状態を管理する。具体的には、仮想資源管理部２８は、アクセス要求が仮想資源を削除するものであった場合には、該仮想資源に対応する削除フラグを設定する。これにより、ファイルの実態は残しつつ、ファイルの参照を不可にすることで、整合性を保つ。

【００４８】

ここで、変換テーブルとは、図５に例示するように、資源のファイルパスを示す「実ファイルパス」と仮想資源のファイルパスを示す「仮想ファイルパス」とアクセス可能か否かを示す「フラグ」とを対応付けて記憶する。そして、仮想資源管理部２８は、アクセス要求が仮想資源を削除するものであった場合には、図５に例示するように、変換テーブルに対して、該仮想資源に対応する仮想ファイルパスに対応するフラグに「アクセス不可」を設定する。

【００４９】

つまり、従来であれば、仮想的な隔離環境において、資源を新たに作成しようとした後に当該ホスト資源を削除することで、解析終了時において仮想資源の実体が保存できない問題があった。例えば、１．File_Aを作成し、２．File_Aを削除してプログラムが終了した場合に、終了時にすでにFile_Aが削除されており存在しないため、作成されたFile_Aが保存できない。もしくは、資源の削除を許可しないアクセス要求制御を行う場合は、資源

削除のアクセス要求が失敗もしくは削除したはずの資源にアクセス可能なことから、隔離環境で解析していることがプログラムやモジュールに知られてしまう。例えば、１．File_Aを作成し、２．File_AをFile_Bにコピーし、３．File_Aを削除し、４．File_Aが削除されているか確認する、という一連の動作を行う場合において、削除されたはずのFile_Aに対してアクセス可能であることによる資源の不整合から、プログラムに隔離環境であることが把握されてしまう。

【００５０】

そこで、本実施例では、仮想的な隔離環境において、資源を新たに作成した後に当該資源を削除することで、解析終了時において資源の実体が保存できない問題を解決するために、アクセス要求が資源を削除するものである場合に、当該資源と同一の仮想資源が存在すればその仮想資源に対して削除フラグを付与することで以降の当該仮想資源に対するアクセスを出来なくする。

【００５１】

また、変換テーブルは、ファイルシステムやメモリ上の任意の場所（例えば特定のファイルパスや共有メモリのアドレス）で管理される。当該プログラムもしくはモジュール毎に異なる資源テーブルを利用するものであり、他のプロセスの変換テーブルは利用できない。なぜならば、プロセス毎に資源テーブルの管理場所が異なり、その場所は他のプロセスからは推測ができないからである。

【００５２】

また、仮想資源管理部２８は、アクセス要求が仮想資源をコピーするものであった場合には、該仮想資源のコピーを行う代わりにコピー先の仮想資源として、既に作成された仮想資源を対応付けて変換テーブルに管理する。つまり、仮想ファイル自体は新たに生成せず、実ファイルと仮想ファイルの対応情報だけテーブルに登録することで、ファイルの実体が増えすぎることを防止し、ファイル保存容量の削減を行うことができる。

【００５３】

例えば、従来では、頻繁にコピーが発生する場合には、図６の（１）に示すように、ファイルの実体がコピーした分だけ生成されるが、本実施例では、図６の（２）に示すように、仮想ファイル自体は新たに生成せず、実ファイルと仮想ファイルの対応情報だけテーブルに登録する。これにより、図６（２）の例では、ファイルは実体として１ファイル（D:\secure\abcd…）のみとなり、ファイル保存容量の削減を行うことができる。

【００５４】

次に、図７～９を用いて、上述した各機能部によって行われる処理の詳細について説明する。まず、図７を用いて、プログラム実行部２１によって行われる処理の処理手順について説明する。図７は、プログラム実行部２１によって行われる処理の処理手順を示すフローチャートである。

【００５５】

なお、ここでは、監視システムの利用者に監視対象プログラム３０を指定させることにより、監視部２０上で監視対象プログラム３０を起動させる場合について説明する。

【００５６】

図７に示すように、この場合には、プログラム実行部２１は、利用者によって監視対象プログラム３０が指定されると（ステップＳ１０１，Ｙｅｓ）、続いて、指定された監視対象プログラム３０全体を監視するか否かを利用者に選択させる（ステップＳ１０２）。つまり、プログラム実行部２１は、指定された監視対象プログラム３０全体を監視するか監視対象プログラム３０を構成するモジュール単位で監視するかを利用者に選択させる。

【００５７】

そして、モジュール単位で監視すると選択された場合には（ステップＳ１０２，Ｙｅｓ）、プログラム実行部２１は、さらに、利用者に監視対象のモジュールを指定させる（ステップＳ１０３）。

【００５８】

ここで、監視対象のモジュールが指定された場合（ステップＳ１０３，Ｙｅｓ）、また

は、監視対象プログラム３０全体を監視すると選択されていた場合（ステップＳ１０２，Ｎｏ）には、プログラム実行部２１は、すでに指定されている監視対象プログラム３０に対して監視用モジュール（資源アクセス監視部２２、資源アクセス制御部２３、仮想資源処理部２５、プロセス監視部２６、プロセス制御部２７、仮想資源管理部２８）を挿入する（ステップＳ１０４）。

【００５９】

そして、プログラム実行部２１は、監視用モジュールを挿入した監視対象プログラム３０を監視部２０上で起動させる（ステップＳ１０５）。

【００６０】

次に、図８を用いて、資源アクセス監視部２２、資源アクセス制御部２３、仮想資源処理部２５、プロセス監視部２６、プロセス制御部２７、仮想資源管理部２８によって行われる処理の処理手順について説明する。図８は、資源アクセス監視部２２、資源アクセス制御部２３、仮想資源処理部２５、プロセス監視部２６、プロセス制御部２７、仮想資源管理部２８によって行われる処理の処理手順を示すフローチャートである。なお、ここで説明する処理手順は、図７に示した処理手順で監視対象プログラム３０が起動された後に実行される。

【００６１】

同図に示すように、資源アクセス監視部２２が、ホスト資源へのアクセス要求を検知した場合には（ステップＳ２０１，Ｙｅｓ）、資源アクセス制御部２３が、当該アクセス要求の種類に基づいて、当該アクセス要求がホスト資源に影響を与えるものであるか否かを判定する（ステップＳ２０２）。

【００６２】

そして、資源アクセス制御部２３によって、アクセス要求がホスト資源に影響を与えるものであると判定された場合には（ステップＳ２０２，Ｙｅｓ）、仮想資源処理部２５が、そのアクセス要求がホスト資源を新規に作成するものであるか否かを判定する（ステップＳ２０３）。

【００６３】

そして、アクセス要求がホスト資源を新規に作成するものであった場合には（ステップＳ２０３，Ｙｅｓ）、仮想資源処理部２５が、所定の格納場所に、当該ホスト資源に対応する仮想資源を新規に作成する（ステップＳ２０４）。

【００６４】

例えば、アクセス要求が、ホストシステムのＣドライブにtest.txtというファイルを作成する事を要求するものであったとする。その場合には仮想資源処理部２５は、例えばＤドライブの「Secure」フォルダにtest.txtと同じ内容のvirtual_test.txtというファイルを作成する。そして仮想資源処理部２５は、作成したvirutal_test.txtに監視対象プログラムをアクセスさせる。この場合、C:\test.txtとD:\Secure\virtual_test.txtとを対応付けた情報が動作ログ保存部２４に作成される。なお、仮想資源のファイル名や作成する場所は任意の場所を設定することができる。

【００６５】

その後、アクセス対象の仮想資源に削除フラグが付与されているか判定する（ステップＳ２１１）。この結果、仮想資源に削除フラグが付与されていると判定された場合には（ステップＳ２１１，Ｙｅｓ）、ホスト資源へのアクセスを拒否し（ステップＳ２１５）、仮想資源処理部２５は、それぞれのアクセスの履歴を動作ログとして動作ログ保存部２４に保存する（ステップＳ２１６）。

【００６６】

また、仮想資源に削除フラグが付与されていないと判定された場合には（ステップＳ２１１，Ｎｏ）、アクセス要求がホスト資源をコピーするものであるか判定する（ステップＳ２１２）。この結果、アクセス要求がホスト資源をコピーするものである場合には（ステップＳ２１２，Ｙｅｓ）、仮想資源管理部２８が、コピー先のホスト資源として、すでに作成されている仮想資源を対応付けて管理する（ステップＳ２１３）。そして、仮想資

源処理部２５は、それぞれのアクセスの履歴を動作ログとして動作ログ保存部２４に保存する（ステップＳ２１６）。

【００６７】

また、アクセス要求がホスト資源をコピーするものでない場合には（ステップＳ２１２，Ｎｏ）、仮想資源処理部２５は、アクセス要求によって要求されたホスト資源の代わりに、すでに作成されている仮想資源に監視対象プログラム３０をアクセスさせる（ステップＳ２１４）。そして、仮想資源処理部２５は、それぞれのアクセスの履歴を動作ログとして動作ログ保存部２４に保存する（ステップＳ２１６）。

【００６８】

Ｓ２０２に戻って、資源アクセス制御部２３によって、アクセス要求がホスト資源に影響を与えるものでないと判定された場合には（ステップＳ２０２，Ｎｏ）、仮想資源処理部２５は、動作ログ保存部２４に保存されている動作ログを参照して、アクセスを要求されたホスト資源に対応する仮想資源が作成されているか否かを判定する（ステップＳ２０９）。

【００６９】

そして、仮想資源が作成されていた場合には（ステップＳ２０９，Ｙｅｓ）、上記したステップＳ２１１に進む。一方、仮想資源が作成されていなかった場合には（ステップＳ２０９，Ｎｏ）、仮想資源処理部２５は、アクセスを要求されたホスト資源に監視対象プログラム３０をアクセスさせる（ステップＳ２１０）。

【００７０】

Ｓ２０３に戻って、アクセス要求がホスト資源を新規に作成するものでなかった場合には（ステップＳ２０３，Ｎｏ）、仮想資源処理部２５は、動作ログ保存部２４に保存されている動作ログを参照し、アクセスを要求されたホスト資源に対応する仮想資源が過去に作成されているか否かを判定する（ステップＳ２０５）。

【００７１】

そして、仮想資源が作成されていた場合には（ステップＳ２０５，Ｙｅｓ）、ステップＳ２０７に進む。一方、仮想資源が作成されていなかった場合には（ステップＳ２０５，Ｎｏ）、仮想資源処理部２５は、アクセスを要求されたホスト資源（たとえば、ファイルなど）を所定の格納場所にコピーし、コピーした資源を仮想資源とする（ステップＳ２０６）。

【００７２】

続いて、アクセス要求がホスト資源を削除するものであるか判定する（ステップＳ２０７）。この結果、アクセス要求がホスト資源を削除するものでないと判定された場合には（ステップＳ２０７，Ｎｏ）、上記のステップＳ２１１に進む。

【００７３】

また、アクセス要求がホスト資源を削除するものであると判定された場合には（ステップＳ２０７，Ｙｅｓ）、仮想資源に削除フラグを立てる（ステップＳ２０８）。これにより、以降のアクセス要求を許可しない。例えば、C:\Windows（登録商標）\system32\calc.exeのファイルを改変し、改変したファイルを削除した場合は、仮想資源として作成した「改変されたC:\Windows（登録商標）\system32\calc.exeファイル」に対して以降アクセスすることができない。ただし、アクセス要求が存在するホスト資源を作成する場合、すでに仮想資源に当該ホスト資源と同一に作られたもので削除フラグが付与されているかどうかを確認し、削除フラグが付いている場合は、ホスト資源へのアクセスを拒否する。

【００７４】

その後、仮想資源処理部２５は、それぞれのアクセスの履歴を動作ログとして動作ログ保存部２４に保存する（ステップＳ２１６）。

【００７５】

次に、図９を用いて、プロセス制御部２７によって行われる処理の処理手順について説明する。図９は、プロセス制御部によって行われる処理の処理手順を示すフローチャートである。

【００７６】
同図に示すように、プロセス制御部２７は、監視対象プログラム３０からプロセス生成の要求が検知されると（ステップＳ３０１，Ｙｅｓ）、生成するプロセスがプロセス要求許可リストに存在するか否かを判定する（ステップＳ３０２）。この結果、プロセス制御部２７は、生成するプロセスがプロセス要求許可リストに存在しないと判定した場合には（ステップＳ３０２，Ｎｏ）、プロセス要求を拒否し（ステップＳ３０４）、動作ログを保存して（ステップＳ３０９）、処理を終了する。
【００７７】
また、プロセス制御部２７は、生成するプロセスがプロセス要求許可リストに存在すると判定した場合には（ステップＳ３０２，Ｙｅｓ）、指定された監視対象プログラム３０全体を監視するか否かを利用者に選択させる（ステップＳ３０３）。つまり、プロセス制御部２７は、指定された監視対象プログラム３０全体を監視するか監視対象プログラム３０を構成するモジュール単位で監視するかを利用者に選択させる。
【００７８】
そして、モジュール単位で監視すると選択された場合には（ステップＳ３０３，Ｙｅｓ）、プロセス制御部２７は、さらに、利用者に監視対象のモジュールを指定させる（ステップＳ３０５）。
【００７９】
ここで、監視対象のモジュールが指定された場合（ステップＳ３０５，Ｙｅｓ）、または、監視対象プログラム３０全体を監視すると選択されていた場合（ステップＳ３０３，Ｎｏ）には、すでに指定されている監視対象プログラム３０に対して監視用モジュール（資源アクセス監視部２２、資源アクセス制御部２３、仮想資源処理部２５、プロセス監視部２６、プロセス制御部２７、仮想資源管理部２８）を挿入する（ステップＳ３０６）。
【００８０】
そして、プロセス制御部２７は、変換テーブルを、作成したプロセスに通知する（ステップＳ３０７）。具体的には、プロセス制御部２７は、作成したプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定し、関連するプロセスであると判定された場合には、変換テーブルを、作成したプロセスに通知する。
【００８１】
そして、プロセス制御部２７は、監視用モジュールを挿入した監視対象プログラム３０を監視部２０上で起動させ（ステップＳ３０８）、それぞれのアクセスの履歴を動作ログとして動作ログ保存部２４に保存し（ステップＳ３０９）、処理を終了する。
【００８２】
[実施例の効果]
上述してきたように、監視部２０は、プロセスの生成要求が検知された場合に、当該プロセスの生成要求に基づいて生成されるプロセスが、既に生成されたプロセスと関連する関連プロセスであるか否かを判定する。そして、監視部２０は、関連するプロセスであると判定された場合には、既に生成されたプロセスに対応する仮想資源のアドレスを関連プロセスに通知し、該仮想資源を関連プロセスが参照するように制御する。このため、プロセスを仮想的な隔離環境で実行した場合においても、異なるプロセス間で資源を共有することが可能である。
【００８３】
つまり、異なるプログラムやモジュールが共通の資源を参照して動作をするものであった場合、各プログラムやモジュール毎に作成された仮想資源のみを参照することによる資源の見え方の違いから二者間で不整合が発生するが、プログラムやモジュールの関連性を把握したうえで関連性があると判断できた場合は共通の仮想資源を参照させるため、協調動作を行う際の異なるプログラムもしくはモジュール間における資源の不整合を解消することができる。
【００８４】
また、監視部２０は、プロセス生成の要求に基づいて生成されたプロセスが、既に生成

されたプロセスの子プロセスであるか否かを判定し、子プロセスであると判定された場合には、既に生成されたプロセスに対応する仮想資源のアドレスを子プロセスに通知し、該仮想資源を子プロセスが参照するように制御する。このため、親プロセスと子プロセスとの間で資源を共有することが可能である。

【００８５】

また、監視部２０は、アクセス要求が仮想資源を削除するものであった場合には、該仮想資源に対応する削除フラグを設定し、削除フラグが設定された仮想資源へのアクセスがあった場合には、該アクセスを拒否する。このため、資源の実体を残しつつ、ファイルの参照を不可能にすることで、整合性を保つことが可能である。

【００８６】

また、監視部２０は、アクセス要求が仮想資源をコピーするものであった場合には、該仮想資源のコピーを行う代わりにコピー先の仮想資源として、既に作成された仮想資源を対応付けて管理する。このため、資源の複製を行った場合に、仮想資源としては単一のものとして管理するため、仮想資源を確保するための保存領域を削減することが可能である。

【００８７】

ところで、上記実施例では、監視部２０をホストシステム１０上で動作させる場合について説明した。しかしながら、本発明に係る監視システムの実施形態はこれに限られるわけではなく、他のさまざまな形態でも実施が可能である。

【００８８】

例えば、監視部２０の一部をホストシステム１０の一部として動作させてもよい。この場合、具体的には、監視部２０が有するモジュールのうち、１つまたは２つ以上のモジュールを、カーネルモジュールとしてホストシステム１０内で動作させる。

【００８９】

ここで、監視部２０が有するモジュールを全てホストシステム１０内で動作させるか、一部を動作させるかは、システム全体の実装の形態に依存する。そして、監視対象プログラム３０は、ホストシステム１０上、または、監視部２０の一部の上で動作させる。

【００９０】

これまでに説明してきたように、監視システムによれば、１つのプログラムを解析するのに必要なコストを低く抑えることができる。また、実環境に近い環境で監視対象プログラム３０を動作させることができるので、監視対象プログラムの動作を阻害することがない。したがって、大量かつ正確にプログラムを解析することが可能になる。

【００９１】

かかる監視システムは、コンピュータウィルスやワームといった悪性プログラムの解析や、脆弱性を有するプログラムなどの信頼できないプログラムの解析などに利用することができる。また、脆弱性を有する監視システム上で動作させることによって、ハニーポットとして利用することも可能である。この他、軽量な仮想マシンとして利用するなど、監視システムは、ここで説明した例に限られず、他にも各種の形態で利用することが可能である。

【００９２】

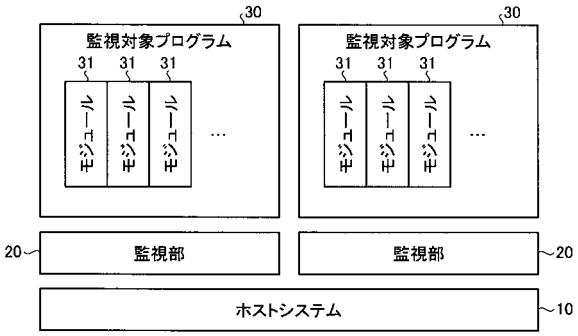
［プログラム］

また、上記実施形態において説明した監視部２０が実行する処理をコンピュータが実行可能な言語で記述したプログラムを作成することもできる。例えば、監視部２０が実行する処理をコンピュータが実行可能な言語で記述した監視プログラムを作成することもできる。この場合、コンピュータが監視プログラムを実行することにより、上記実施形態と同様の効果を得ることができる。さらに、かかる監視プログラムをコンピュータ読み取り可能な記録媒体に記録して、この記録媒体に記録された監視プログラムをコンピュータに読み込ませて実行することにより上記第一の実施形態と同様の処理を実現してもよい。以下に、図２に示した監視部２０と同様の機能を実現する監視プログラムを実行するコンピュータの一例を説明する。

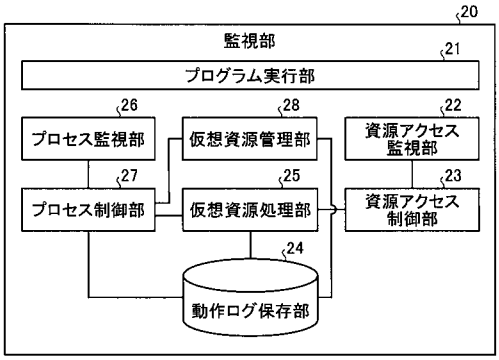
【００９３】
図１０は、監視プログラムを実行するコンピュータ１０００を示す図である。図１０に例示するように、コンピュータ１０００は、例えば、メモリ１０１０と、ＣＰＵ１０２０と、ハードディスクドライブインタフェース１０３０と、ディスクドライブインタフェース１０４０と、シリアルポートインタフェース１０５０と、ビデオアダプタ１０６０と、ネットワークインタフェース１０７０とを有し、これらの各部はバス１０８０によって接続される。
【００９４】
メモリ１０１０は、図１０に例示するように、ＲＯＭ（Read Only Memory）１０１１及びＲＡＭ１０１２を含む。ＲＯＭ１０１１は、例えば、ＢＩＯＳ（Basic Input Output System）等のブートプログラムを記憶する。ハードディスクドライブインタフェース１０３０は、図１０に例示するように、ハードディスクドライブ１０３１に接続される。ディスクドライブインタフェース１０４０は、図１０に例示するように、ディスクドライブ１０４１に接続される。例えば磁気ディスクや光ディスク等の着脱可能な記憶媒体が、ディスクドライブに挿入される。シリアルポートインタフェース１０５０は、図１０に例示するように、例えばマウス１０５１、キーボード１０５２に接続される。ビデオアダプタ１０６０は、図１０に例示するように、例えばディスプレイ１０６１に接続される。
【００９５】
ここで、図１０に例示するように、ハードディスクドライブ１０３１は、例えば、ＯＳ１０９１、アプリケーションプログラム１０９２、プログラムモジュール１０９３、プログラムデータ１０９４を記憶する。すなわち、上記の監視プログラムは、コンピュータ１０００によって実行される指令が記述されたプログラムモジュールとして、例えばハードディスクドライブ１０３１に記憶される。
【００９６】
また、上記実施形態で説明した各種データは、プログラムデータとして、例えばメモリ１０１０やハードディスクドライブ１０３１に記憶される。そして、ＣＰＵ１０２０が、メモリ１０１０やハードディスクドライブ１０３１に記憶されたプログラムモジュール１０９３やプログラムデータ１０９４を必要に応じてＲＡＭ１０１２に読み出し、アクセス監視手順、アクセス制御手順、プロセス監視手順、プロセス制御手順を実行する。
【００９７】
なお、監視プログラムに係るプログラムモジュール１０９３やプログラムデータ１０９４は、ハードディスクドライブ１０３１に記憶される場合に限られず、例えば着脱可能な記憶媒体に記憶され、ディスクドライブ等を介してＣＰＵ１０２０によって読み出されてもよい。あるいは、監視プログラムに係るプログラムモジュール１０９３やプログラムデータ１０９４は、ネットワーク（ＬＡＮ（Local Area Network）、ＷＡＮ（Wide Area Network）等）を介して接続された他のコンピュータに記憶され、ネットワークインタフェース１０７０を介してＣＰＵ１０２０によって読み出されてもよい。
【符号の説明】
【００９８】
１０　ホストシステム
２０　監視部
２１　プログラム実行部
２２　資源アクセス監視部
２３　資源アクセス制御部
２４　動作ログ保存部
２５　仮想資源処理部
２６　プロセス監視部
２７　プロセス制御部
２８　仮想資源管理部
３０　監視対象プログラム

３１　モジュール

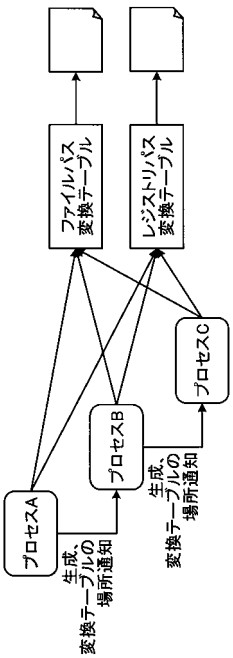
【図１】



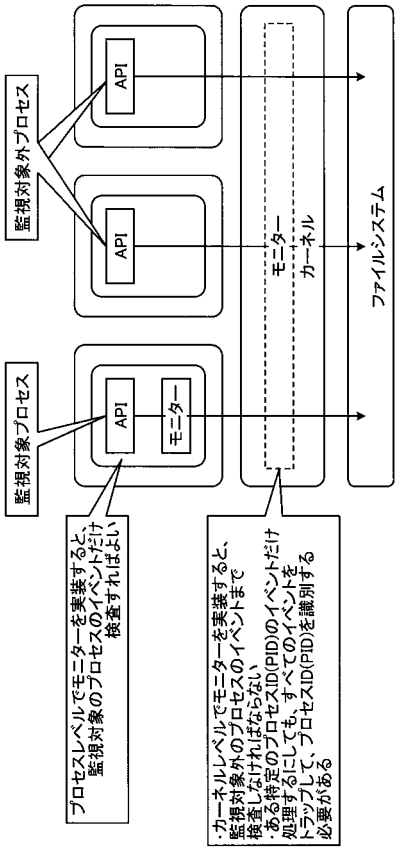
【図２】



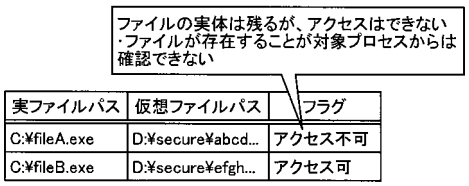
【図３】



【図４】

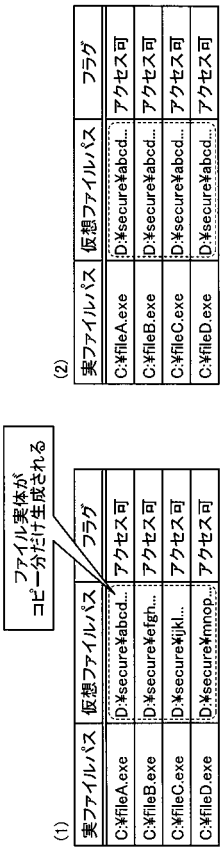


【図５】



実ファイルパス	仮想ファイルパス	フラグ
C:¥fileA.exe	D:¥secure¥abcd...	アクセス不可
C:¥fileB.exe	D:¥secure¥efgh...	アクセス可

【図６】



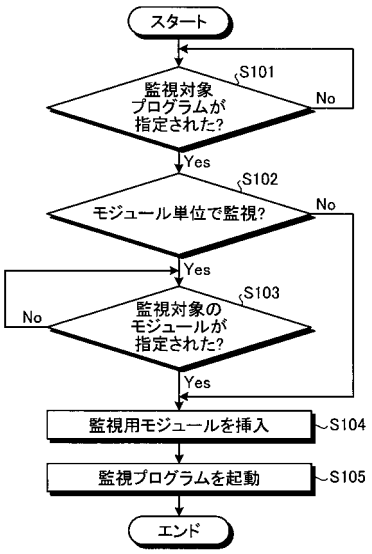
(1)

実ファイルパス	仮想ファイルパス	フラグ
C:¥fileA.exe	D:¥secure¥abcd...	アクセス可
C:¥fileB.exe	D:¥secure¥efgh...	アクセス可
C:¥fileC.exe	D:¥secure¥ijkl...	アクセス可
C:¥fileD.exe	D:¥secure¥mnop...	アクセス可

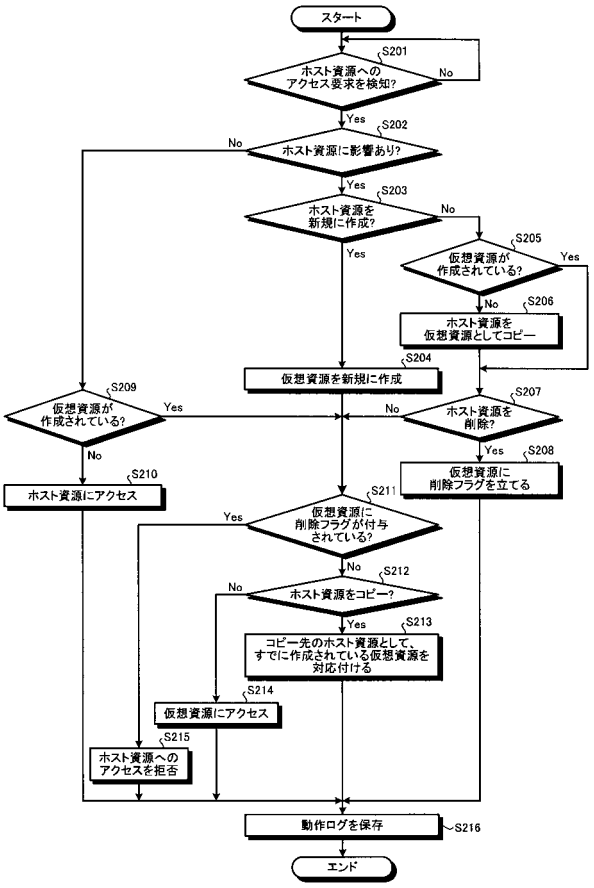
(2)

実ファイルパス	仮想ファイルパス	フラグ
C:¥fileA.exe	D:¥secure¥abcd...	アクセス可
C:¥fileB.exe	D:¥secure¥abcd...	アクセス可
C:¥fileC.exe	D:¥secure¥abcd...	アクセス可
C:¥fileD.exe	D:¥secure¥abcd...	アクセス可

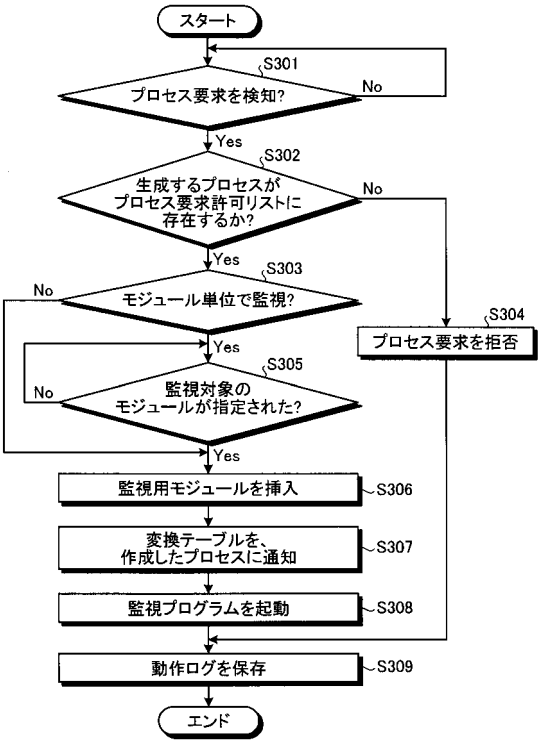
【図７】



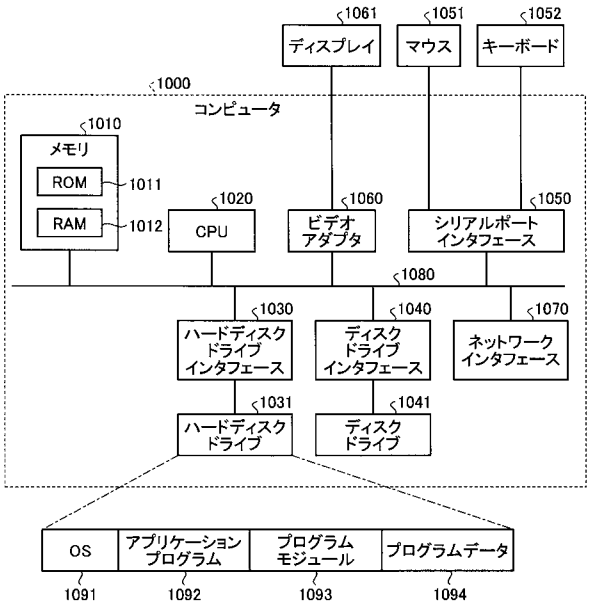
【図８】


スタート
S201
ホスト資源へのアクセス要求を検知?
No
Yes
S202
ホスト資源に影響あり?
No
Yes
S203
ホスト資源を新規に作成?
No
Yes
S205
仮想資源が作成されている?
Yes
No
S206
ホスト資源を仮想資源としてコピー
S204
仮想資源を新規に作成
S209
仮想資源が作成されている?
Yes
No
S207
ホスト資源を削除?
No
Yes
S208
仮想資源に削除フラグを立てる
S210
ホスト資源にアクセス
S211
仮想資源に削除フラグが付与されている?
Yes
No
S212
ホスト資源をコピー?
No
Yes
S213
コピー先のホスト資源として、すでに作成されている仮想資源を対応付ける
S214
仮想資源にアクセス
S215
ホスト資源へのアクセスを拒否
動作ログを保存
S216
エンド

【図９】


スタート
S301
プロセス要求を検知?
No
Yes
S302
生成するプロセスがプロセス要求許可リストに存在するか?
No
Yes
S303
モジュール単位で監視?
No
Yes
S304
プロセス要求を拒否
S305
監視対象のモジュールが指定された?
No
Yes
監視用モジュールを挿入
S306
変換テーブルを、作成したプロセスに通知
S307
監視プログラムを起動
S308
動作ログを保存
S309
エンド

【図１０】


1061
ディスプレイ
1051
マウス
1052
キーボード
1000
コンピュータ
1010
メモリ
ROM
1011
RAM
1012
1020
CPU
1060
ビデオアダプタ
1050
シリアルポートインタフェース
1080
1030
ハードディスクドライブインタフェース
1040
ディスクドライブインタフェース
1070
ネットワークインタフェース
1031
ハードディスクドライブ
1041
ディスクドライブ
OS
アプリケーションプログラム
プログラムモジュール
プログラムデータ
1091
1092
1093
1094

フロントページの続き

(56)参考文献 特開２０００－２０７１３（ＪＰ，Ａ）
特開２００５－３３２１１０（ＪＰ，Ａ）
特開２０１０－２６５７２（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　２１／５３
Ｇ０６Ｆ　９／４６
Ｇ０６Ｆ　９／５４