(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2004-139177
(P2004-139177A)
(43)公開日　平成16年5月13日(2004.5.13)

(51)Int.Cl.[7]	F I			テーマコード（参考）
G06F 1/00	G06F	9/06	660J	5B076
G06F 11/00	G06F	9/06	660N	

審査請求 未請求　請求項の数 13　OL　（全 11 頁）

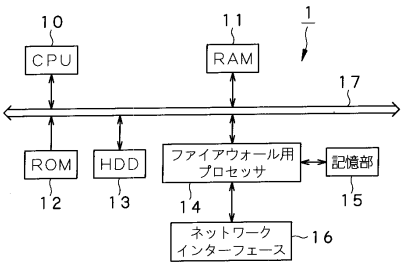
(21)出願番号　特願2002-300813(P2002-300813)
(22)出願日　平成14年10月15日(2002.10.15)

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番３５号
(74)代理人　100067736
弁理士　小池　晃
(74)代理人　100086335
弁理士　田村　榮一
(74)代理人　100096677
弁理士　伊賀　誠司
(72)発明者　齊藤　真也
東京都品川区北品川６丁目７番３５号　ソニー株式会社内

Fターム(参考)　5B076 FD00　FD08

(54)【発明の名称】情報検査方法及び装置、並びにプログラム

(57)【要約】
【課題】入力情報をフィルタリングするためのルールセットの増加を防止し、フィルタリングの効率を向上させると共に、必要となる記憶メモリを削減する。
【解決手段】情報処理装置１において、ファイアウォール用プロセッサ１４は、ソフトウェア上の脆弱性とこの脆弱性に起因するセキュリティ上の攻撃、例えば権限のないアクセスやウイルスを阻止するためのルールセットとを対応付けて記憶部１５に記憶保持している。そして、少なくとも上記所定のソフトウェアを修正するためのデータと、該所定のソフトウェア上の脆弱性を示す識別子とにより構成される修正プログラムによってこの脆弱性が解決された場合に、対応する不要となったルールを削除する。
【選択図】　図１



【特許請求の範囲】
【請求項１】
所定の問題と該問題に起因するセキュリティ上の攻撃を阻止するためのルールとが対応付けられて記憶された記憶手段を用い、入力情報に対して上記ルールを適用してフィルタリングを行うことによりセキュリティポリシーを実現するフィルタリング工程と、
上記問題が解決された場合に、該問題と対応付けられたルールを削除して上記記憶手段を更新する更新工程と
を有することを特徴とする情報検査方法。
【請求項２】
上記所定の問題は、所定のソフトウェア上の脆弱性であることを特徴とする請求項１記載の情報検査方法。
【請求項３】
上記セキュリティ上の攻撃は、上記脆弱性を有する上記所定のソフトウェアが利用するポートを介した攻撃であることを特徴とする請求項２記載の情報検査方法。
【請求項４】
上記セキュリティ上の攻撃は、上記脆弱性を利用するコンピュータウイルスによる攻撃であることを特徴とする請求項２記載の情報検査方法。
【請求項５】
上記更新工程では、所定の修正プログラムによって上記脆弱性が解決された場合に、該脆弱性と対応付けられたルールが削除され、上記記憶手段が更新されることを特徴とする請求項２記載の情報検査方法。
【請求項６】
上記所定の修正プログラムは、少なくとも上記所定のソフトウェアを修正するためのデータと、該所定のソフトウェア上の脆弱性を示す識別子とにより構成されることを特徴とする請求項５記載の情報検査方法。
【請求項７】
所定の問題と該問題に起因するセキュリティ上の攻撃を阻止するためのルールとが対応付けられて記憶された記憶手段と、
入力情報に対して上記ルールを適用してフィルタリングを行うことによりセキュリティポリシーを実現するフィルタリング手段とを備え、
上記フィルタリング手段は、上記問題が解決された場合に、該問題と対応付けられたルールを削除すること
を特徴とする情報検査装置。
【請求項８】
上記所定の問題は、所定のソフトウェア上の脆弱性であることを特徴とする請求項７記載の情報検査装置。
【請求項９】
上記セキュリティ上の攻撃は、上記脆弱性を有する上記所定のソフトウェアが利用するポートを介した攻撃であることを特徴とする請求項８記載の情報検査装置。
【請求項１０】
上記セキュリティ上の攻撃は、上記脆弱性を利用するコンピュータウイルスによる攻撃であることを特徴とする請求項８記載の情報検査装置。
【請求項１１】
上記フィルタリング手段は、所定の修正プログラムによって上記脆弱性が解決された場合に、該脆弱性と対応付けられたルールを削除することを特徴とする請求項８記載の情報検査装置。
【請求項１２】
上記所定の修正プログラムは、少なくとも上記所定のソフトウェアを修正するためのデータと、該所定のソフトウェア上の脆弱性を示す識別子とにより構成されることを特徴とする請求項１１記載の情報検査装置。

【請求項１３】
所定の処理をコンピュータに実行させるプログラムにおいて、
所定の問題と該問題に起因するセキュリティ上の攻撃を阻止するためのルールとが対応付けられて記憶された記憶手段を用い、入力情報に対して上記ルールを適用してフィルタリングを行うことによりセキュリティポリシーを実現するフィルタリング工程と、
上記問題が解決された場合に、該問題と対応付けられたルールを削除して上記記憶手段を更新する更新工程と
を有することを特徴とするプログラム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、コンピュータネットワークにおける権限のないアクセス又はコンピュータウイルスを防止するために、ルールセットを用いて入力情報のフィルタリングを行う情報検査方法及びその装置、並びにプログラムに関する。
【０００２】
【従来の技術】
コンピュータネットワークでは、通常、パケットの形で情報が伝送される。そして、あるサイトにある情報は、他のサイトからのコマンドによりアクセスされ、又は他のサイトへと伝送される。したがって、価値のある情報を有している場合、その情報へのアクセスについては権限の有無を検査し、権限のないアクセスを防止する必要がある。
【０００３】
また、コンピュータウイルスのように情報の価値を破壊するような情報も存在するため、入力パケットがコンピュータウイルスであるか否かを検査する必要もある。
【０００４】
いわゆるファイアウォールやアンチウイルスとして知られるソフトウェアは、前述の検査を行うソフトウェアである。通常、これらのソフトウェアはルールセットを記憶しており、このルールセットに従ってアクセスを許可するか拒否するかのフィルタリング処理を行う。
【０００５】
ここで、高度なセキュリティポリシーを実現するためには、多数のルールセットが必要となるため、これらのルールセットを用いた入力パケットの検査に時間がかかり、フィルタリングの効率が低下するという問題があった。
【０００６】
そこで、以下に示す特許文献１では、所定の順序で適用されるルールセットを、例えば入力パケットとマッチした回数に応じて再配列する技術が提案されている。この特許文献１に記載の技術によれば、入力パケットが早期にルールセットにマッチするようになるため、フィルタリングの効率を向上させることができる。
【０００７】
【特許文献１】
特開２０００－１７４８０８号公報
【０００８】
【発明が解決しようとする課題】
しかしながら、コンピュータネットワーク上のセキュリティ攻撃やコンピュータウイルスによる攻撃は、時間経過と共に増加しており、それに対抗するためにルールセットも時間経過と共に増加の一途を辿っている。
【０００９】
その結果、ルールセットは肥大化を続け、記憶メモリの消費や検査にかかる時間の増大といった問題が発生しているが、このような問題は、上記特許文献１のようなルールセットの再配列によっては、完全に解決することはできない。
【００１０】

本発明は、このような従来の実情に鑑みて提案されたものであり、入力情報をフィルタリングするためのルールセットの増加を防止し、フィルタリングの効率を向上させると共に、必要となる記憶メモリを削減する情報検査方法及びその装置、並びにその情報検査処理をコンピュータに実行させるプログラムを提供することを目的とする。

【００１１】

【課題を解決するための手段】

上述した目的を達成するために、本発明に係る情報検査方法及びその装置は、所定の問題と該問題に起因するセキュリティ上の攻撃を阻止するためのルールとが対応付けられて記憶された記憶手段を用い、入力情報に対して上記ルールを適用してフィルタリングを行うことによりセキュリティポリシーを実現し、上記問題が解決された場合に、該問題と対応付けられたルールを削除して上記記憶手段を更新する。

【００１２】

ここで、上記所定の問題は、所定のソフトウェア上の脆弱性であり、上記セキュリティ上の攻撃は、上記脆弱性を有する上記所定のソフトウェアが利用するポートを介した攻撃、又は上記脆弱性を利用するコンピュータウイルスによる攻撃である。

【００１３】

また、本情報検査方法及びその装置では、所定の修正プログラムによって上記脆弱性が解決された場合に、該脆弱性と対応付けられたルールが削除され、上記記憶手段が更新される。

【００１４】

また、本発明に係るプログラムは、このような情報検査処理をコンピュータに実行させるものである。

【００１５】

【発明の実施の形態】

以下、本発明を適用した具体的な実施の形態について、図面を参照しながら詳細に説明する。この実施の形態は、本発明を、ネットワークと接続され、権限のないアクセス又はコンピュータウイルス（以下、単にウイルスという。）に対するフィルタリング処理を行うファイアウォール機能を有する情報処理装置に適用したものである。

【００１６】

本実施の形態における情報処理装置の概略構成を図１に示す。図１に示すように、情報処理装置１は、該情報処理装置１の各部を統括して制御するＣＰＵ（Ｃｅｎｔｒａｌ Ｐｒｏｃｅｓｓｉｎｇ Ｕｎｉｔ）１０と、各種情報を一時記憶するワークエリアとしてのＲＡＭ（Ｒａｎｄｏｍ Ａｃｃｅｓｓ Ｍｅｍｏｒｙ）１１と、各種処理を行うためのプログラム等の各種情報を記憶する読み出し専用のＲＯＭ（Ｒｅａｄ Ｏｎｌｙ Ｍｅｍｏｒｙ）１２と、図示しないハードディスクに対して各種情報を記録及び／又は再生するＨＤＤ（ＨａｒｄＤｉｓｋ Ｄｒｉｖｅ）１３と、上述した権限のないアクセス又はコンピュータウイルスに対するフィルタリング処理を制御するファイアウォール用プロセッサ１４とがバス１７を介して接続されてなる。さらに、ファイアウォール用プロセッサ１４には、後述するパケットフィルタリングのルールセット等が記憶された記憶部１５及びネットワークインターフェース部１６が直接接続されている。

【００１７】

この情報処理装置１において、ファイアウォール用プロセッサ１４は、ネットワークインターフェース部１６を介して入力したパケットのフィルタリング処理を行う。すなわち、後述するように、ファイアウォール用プロセッサ１４は、記憶部１５に記憶保持されたルールセットを用いて入力パケットの検査を行い、通過又は廃棄を決定するフィルタリング処理を行う。

【００１８】

なお、この情報処理装置１は、通常のパーソナルコンピュータであっても、ネットワークとの入出力部に設けられる中継装置であってもよい。

【００１９】

また、本実施の形態では、図１に示すようにメインのＣＰＵ１０とは別にファイアウォール用プロセッサ１４を設けるものとしたが、この構成に限定されるものではなく、通常のコンピュータと同様に、メインのＣＰＵがファイアウォール処理を行うようにしても構わない。

【００２０】

但し、メインのＣＰＵとは別にファイアウォール用プロセッサを設けることで、例えばＤｏＳ攻撃（サービス不能／拒否／妨害攻撃：Ｄｅｎｉａｌ　ｏｆ　Ｓｅｒｖｉｃｅ　Ａｔｔａｃｋ）等の高負荷攻撃を受けた場合であっても、メインのＣＰＵでの処理に影響がでないという利点がある。

【００２１】

以下では、先ず、権限のないアクセスを防止するパケットフィルタリングベースのファイアウォールについて説明する。

【００２２】

あるポートを利用し、ネットワークを介して外部からリクエストを受けるソフトウェアにセキュリティ上の脆弱性（セキュリティホール）がある場合、セキュリティ攻撃の対象となる可能性がある。すなわち例えば、ネットワークを介して非許可者が外部から内部に侵入し、任意のプロセスを管理者権限で実行するという攻撃を受ける可能性がある。

【００２３】

そこで、ファイアウォール用プロセッサ１４は、このソフトウェアの脆弱性によって発生する脅威から守るために、ファイアウォールにパケットフィルタリングのルールを設定すると共に、脆弱性とパケットフィルタリングルールとを対応付けた管理テーブルを記憶部１５に記憶保持している。

【００２４】

この管理テーブルの一例を図２に示す。ここで図２において、「管理番号」は、脆弱性を一元管理する番号を示し、アプリケーション利用側が脆弱性を識別するために利用される。また、「ターゲット識別子」は、脆弱性を識別する番号を示し、アプリケーション提供側が脆弱性を識別するために利用される。また、「パケットフィルタリングルール識別子」は、脆弱性による脅威から守るためのルール（パケットフィルタリングルール）を識別するための情報を示し、「状態識別子」は、その脆弱性が解決／未解決の何れであるかを識別するための情報を示す。

【００２５】

例えば、ターゲット識別子が「Ｖ２０００－００２２」で示される未解決の脆弱性は、パケットフィルタリングルール識別子が「００００３」で示されるパケットフィルタリングルールと対応付けられ、管理番号「００００２」として登録されている。

【００２６】

また、ファイアウォール用プロセッサ１４は、パケットフィルタリングルールの詳しい内容の記載されたテーブルを記憶部１５に記憶保持している。

【００２７】

このテーブルの一例を図３に示す。図３に示すように、各パケットフィルタリングルール毎に、「プロトコル」、送信元及び送信先のアドレスを示す「ソースアドレス」及び「デスティネーションアドレス」、送信元及び送信先のポートを示す「ソースポート」及び「デスティネーションポート」と共に、そのようなパケットが送信されたときの入力の許可／許否を示す「アクション」が設定されている。なお、表中の＊印は、任意であることを示す。

【００２８】

例えば、上述したパケットフィルタリングルール識別子「００００３」で示されるパケットフィルタリングルールは、「プロトコル」、「ソースアドレス」、「ソースポート」及び「デスティネーションアドレス」が任意であり、「デスティネーションポート」が「４５６７」、「アクション」が「許否」として登録されている。すなわち、このパケットフィルタリングルールにより、外部からポート「４５６７」を利用したリクエストは、全て

許否される。

【００２９】

ここで、あるソフトウェアの脆弱性（セキュリティホール）を解決するために、例えば図４に示すような修正プログラムが開発された場合を想定する。図４に示すように、修正プログラムは、例えば、修正プログラムを管理するパッチＩＤ、上述したターゲット識別子ＴＩ、オプションＯＰ、データＤＴにより構成される。なお、オプションＯＰの部分には、その他必要となる情報が記載される。

【００３０】

このような修正プログラムを適用することにより、上記プログラムの脆弱性が解決されると、そのソフトウェアの脆弱性を狙ったセキュリティ攻撃は生じ得ないため、その脆弱性と対応付けられたパケットフィルタリングルールも不要となる。

【００３１】

そこで、本実施の形態における情報処理装置１（図１）は、脆弱性の解決によって不要となったパケットフィルタリングルールをテーブルから削除する処理を行う。

【００３２】

この情報処理装置１の処理を図５のフローチャートを用いて説明する。先ずステップＳ１において、情報処理装置１は、ネットワークインターフェース部１６を介して、図４に示したような修正プログラムを取得する。

【００３３】

次にステップＳ２において、修正プログラムからデータＤＴの部分を抽出し、続くステップＳ３において、データＤＴを適用する。この操作により、アプリケーションに存在していた既知の脆弱性、例えばターゲット識別子「Ｖ２０００－００２２」で示される脆弱性は、完全に解決される。

【００３４】

続いてステップＳ４において、修正プログラムからターゲット識別子ＴＩを抽出し、ステップＳ５において、抽出したターゲット識別子ＴＩと一致するものを図２に示した管理テーブルの「ターゲット識別子」の列から探索する。

【００３５】

ステップＳ６では、管理テーブルに修正プログラムのターゲット識別子ＴＩと一致するものがあったか否かが判別される。一致したものがない場合（Ｎｏ）には、処理を終了し、一致するものがあった場合（Ｙｅｓ）には、ステップＳ７に進む。

【００３６】

ステップＳ７では、管理テーブルのうち、修正プログラムのターゲット識別子ＴＩと一致したターゲット識別子を有する行の「状態識別子」を「未解決」から「解決」に更新する。なお、この処理は、後述するステップＳ１０の処理の後に行うことも可能である。

【００３７】

続いてステップＳ８では、管理テーブルのうち、修正プログラムのターゲット識別子ＴＩと一致したターゲット識別子を有する行の「パケットフィルタリングルール識別子」を抽出する。

【００３８】

ステップＳ９では、抽出したパケットフィルタリングルール識別子と一致するものを図３に示したテーブルの「パケットフィルタリングルール識別子」の列から探索し、ステップＳ１０において、一致した同一行を削除する。この際、削除するのではなく、コメントアウトするようにしても構わない。

【００３９】

次に、ウイルスによる攻撃を防止するアンチウイルスベースのファイアウォールについて説明する。ここで、ウイルスは、汎用ＰＣハードウェア上に実装するための言語を用いて、コンピュータソフトウェアとして実現されている。

【００４０】

あるソフトウェアにセキュリティ上の脆弱性（セキュリティホール）があり、ネットワー

クを介してウイルスが外部から内部へ侵入可能とされている場合、ウイルス攻撃の対象となる可能性がある。すなわち例えば、ネットワークを介してウイルスが外部から内部に侵入し、ソフトウェアの脆弱性、例えばバッファオーバーフローを利用して情報処理装置１内の情報を破壊するという攻撃を受ける可能性がある。

【００４１】

そこで、ファイアウォール用プロセッサ１４は、このソフトウェアの脆弱性によって発生する脅威から守るために、ファイアウォールにアンチウイルスのルールを設定すると共に、脆弱性とウイルスとを対応付けた管理テーブルを記憶部１５に記憶保持している。

【００４２】

この管理テーブルの一例を図６に示す。ここで図６において、「管理番号」は、脆弱性を一元管理する番号を示し、アプリケーション利用側が脆弱性を識別するために利用される。また、「ターゲット識別子」は、脆弱性を識別する番号を示し、アプリケーション提供側が脆弱性を識別するために利用される。また、「ウイルス識別子」は、ウイルスによる脅威から守るためのルール（アンチウイルスルール）を識別するための情報を示し、「状態識別子」は、その脆弱性が解決／未解決の何れであるかを識別するための情報を示す。

【００４３】

例えば、ターゲット識別子が「Ｖ２０００－００１７」で示される未解決の脆弱性は、ウイルス識別子が「００００２」で示されるアンチウイルスルールと対応付けられ、管理番号「００００２」として登録されている。

【００４４】

また、ファイアウォール用プロセッサ１４は、ウイルスの詳しい内容の記載されたテーブルを記憶部１５に記憶保持している。

【００４５】

このテーブルの一例を図７に示す。図７に示すように、各ウイルス毎に、ウイルスのコードの特徴を抽出したウイルス成分データが登録される。

【００４６】

ファイアウォール用プロセッサ１４は、入力パケットのペイロードデータとこのウイルス成分データとの間に関連があるか否かについてパターンマッチングを行うことで、入力パケットの検査を行う。なお、いわゆるアンチウイルスソフトウェアもこのような方法でウイルスを検知している。

【００４７】

例えば、上述したウイルス識別子「００００２」で示されるアンチウイルスルールは、そのウイルス成分データが「ｓｔｒｉｎｇ Ｂ１，ｓｔｒｉｎｇ Ｂ２，ｓｔｒｉｎｇ Ｂ３，ｓｔｒｉｎｇ Ｂ４，・・・」として登録されているため、「ｓｔｒｉｎｇ Ｂ１，ｓｔｒｉｎｇ Ｂ２，ｓｔｒｉｎｇ Ｂ３，ｓｔｒｉｎｇ Ｂ４，・・・」と一致するデータは、ウイルスとして検知され、削除又は検疫等の操作が施される。

【００４８】

ここで、あるソフトウェアの脆弱性（セキュリティホール）を解決するために、例えば図４に示したような修正プログラムが開発された場合を想定する。

【００４９】

このような修正プログラムを適用することにより、上記プログラムの脆弱性が解決されると、そのソフトウェアの脆弱性を狙ったセキュリティ攻撃は生じ得ないため、その脆弱性と対応付けられたアンチウイルスルールも不要となる。

【００５０】

そこで、本実施の形態における情報処理装置１（図１）は、脆弱性の解決によって不要となったアンチウイルスルールをテーブルから削除する処理を行う。

【００５１】

この情報処理装置１の処理を図８のフローチャートを用いて説明する。先ずステップＳ２０において、情報処理装置１は、ネットワークインターフェース部１６を介して、図４に示したような修正プログラムを取得する。

【００５２】
次にステップＳ２１において、修正プログラムからデータＤＴの部分を抽出し、続くステップＳ２２において、データＤＴを適用する。この操作により、アプリケーションに存在していた既知の脆弱性、例えばターゲット識別子「Ｖ２０００－００１７」で示される脆弱性は、完全に解決される。
【００５３】
続いてステップＳ２３において、修正プログラムからターゲット識別子ＴＩを抽出し、ステップＳ２４において、抽出したターゲット識別子ＴＩと一致するものを図６に示した管理テーブルの「ターゲット識別子」の列から探索する。
【００５４】
ステップＳ２５では、管理テーブルに修正プログラムのターゲット識別子ＴＩと一致するものがあったか否かが判別される。一致したものがない場合（Ｎｏ）には、処理を終了し、一致するものがあった場合（Ｙｅｓ）には、ステップＳ２６に進む。
【００５５】
ステップＳ２６では、管理テーブルのうち、修正プログラムのターゲット識別子ＴＩと一致したターゲット識別子を有する行の「状態識別子」を「未解決」から「解決」に更新する。なお、この処理は、後述するステップＳ２９の処理の後に行うことも可能である。
【００５６】
続いてステップＳ２７では、管理テーブルのうち、修正プログラムのターゲット識別子ＴＩと一致したターゲット識別子を有する行の「ウイルス識別子」を抽出する。
【００５７】
ステップＳ２８では、抽出したウイルス識別子と一致するものを図７に示したテーブルの「ウイルス識別子」の列から探索し、ステップＳ２９において、一致した同一行を削除する。この際、削除するのではなく、コメントアウトするようにしても構わない。
【００５８】
以上のように、本実施の形態における情報処理装置１は、ソフトウェア上の脆弱性とこの脆弱性に起因するセキュリティ上の攻撃、例えば権限のないアクセスやウイルスを阻止するためのルールセットとを対応付けて記憶しており、修正プログラム等によってこの脆弱性が解決された場合に、対応する不要となったルールを削除する。
【００５９】
これにより、ルールセットの増加を阻止することができ、その結果、必要となる記憶メモリを削減することができる。また、必要のないルールを適用する必要がないため、フィルタリング効率が改善される。さらに、管理テーブルを参照して必要のないルールを知ることにより、どのような脅威からは既に保護される環境にあるかを知る指針を得ることができる。
【００６０】
なお、本発明は上述した実施の形態のみに限定されるものではなく、本発明の要旨を逸脱しない範囲において種々の変更が可能であることは勿論である。
【００６１】
【発明の効果】
以上詳細に説明したように本発明に係る情報検査方法及びその装置は、所定の問題と該問題に起因するセキュリティ上の攻撃を阻止するためのルールとが対応付けられて記憶された記憶手段を用い、入力情報に対して上記ルールを適用してフィルタリングを行うことによりセキュリティポリシーを実現し、上記問題が解決された場合に、該問題と対応付けられたルールを削除して上記記憶手段を更新する。
【００６２】
ここで、上記所定の問題は、所定のソフトウェア上の脆弱性であり、上記セキュリティ上の攻撃は、上記脆弱性を有する上記所定のソフトウェアが利用するポートを介した攻撃、又は上記脆弱性を利用するコンピュータウイルスによる攻撃である。
【００６３】

また、本情報検査方法及びその装置では、所定の修正プログラムによって上記脆弱性が解決された場合に、該脆弱性と対応付けられたルールが削除され、上記記憶手段が更新される。

【００６４】

このような情報検査方法及びその装置によれば、ソフトウェア上の脆弱性とこの脆弱性に起因するセキュリティ上の攻撃、例えば権限のないアクセスやウイルスを阻止するためのルールセットとを対応付けて記憶し、修正プログラム等によってこの脆弱性が解決された場合に、対応する不要となったルールを削除するため、ルールセットの増加を阻止することができ、その結果、必要となる記憶メモリを削減することができる。また、必要のないルールを適用する必要がないため、フィルタリング効率が改善される。

【００６５】

また、本発明に係るプログラムは、このような情報検査処理をコンピュータに実行させるものである。

【００６６】

このようなプログラムによれば、上述した情報検査処理をソフトウェアにより実現することができる。

【図面の簡単な説明】

【図１】本実施の形態における情報処理装置の概略構成を説明する図である。

【図２】脆弱性とパケットフィルタリングルールとを対応付けた管理テーブルの一例を示す図である。

【図３】パケットフィルタリングルールの詳しい内容の記載されたテーブルの一例を示す図である。

【図４】ソフトウェアの脆弱性を解決する修正プログラムの構成例を示す図である。

【図５】脆弱性の解決によって不要となったパケットフィルタリングルールをテーブルから削除する処理を説明するフローチャートである。

【図６】脆弱性とコンピュータウイルスとを対応付けた管理テーブルの一例を示す図である。

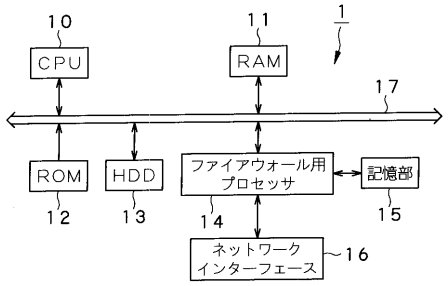
【図７】コンピュータウイルスの詳しい内容の記載されたテーブルの一例を示す図である。

【図８】脆弱性の解決によって不要となったアンチウイルスルールをテーブルから削除する処理を説明するフローチャートである。

【符号の説明】

１　情報処理装置、１０　ＣＰＵ、１１　ＲＡＭ、１２　ＲＯＭ、１３　ＨＤＤ、１４　ファイアウォール用プロセッサ、１５　記憶部、１６　ネットワークインターフェース部、１７　バス

【図１】



【図２】

管理番号	ターゲット識別子	パケットフィルタリング ルール識別子	状態識別子
00001	V2000-0013	00001	解 決
00002	V2000-0022	00003	未解決
00003	V2000-0039	00013	未解決
00004	V2000-0117	00091	未解決

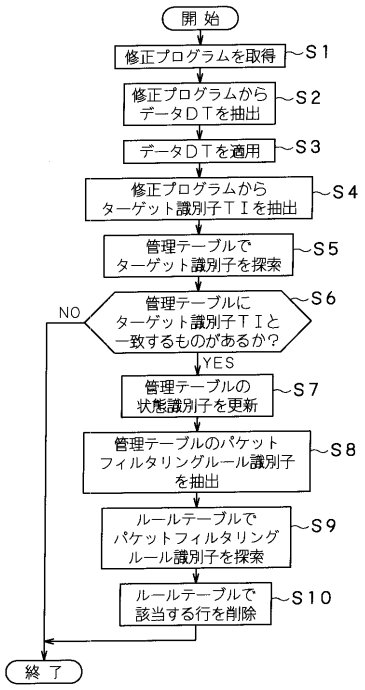
【図３】

パケットフィルタリングルール識別子	プロトコル	ソースアドレス	デスティネーションアドレス	ソースポート	デスティネーションポート	アクション
00001	TCP	123.123.123.123	*	*	*	拒否
00003	*	*	*	*	4567	拒否
……	……	……	……	……	……	……
01230	*	*	*	*	*	許可

【図４】

パッチID	ターゲット識別子ＴＩ	オプションOP	データＤＴ

【図５】



【図６】

管理番号	ターゲット識別子	ウイルス識別子	状態識別子
00001	S2000-013	00001	未解決
00002	S2000-017	00002	未解決
00003	S2000-021	00003	未解決
00004	S2000-033	00004	解 決

【図７】

ウイルス識別子	ウイルス成分データ				
00001	stringA1	stringA2	stringA3	stringA4	・・・・
00002	stringB1	stringB2	stringB3	stringB4	・・・・
00003	stringC1	stringC2	stringC3	stringC4	・・・・
00005	stringD1	stringD2	stringD3	stringD4	・・・・
⋮	⋮	⋮	⋮	⋮	⋮

【図８】