(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2004-199213
(P2004-199213A)
(43)公開日　平成16年7月15日(2004.7.15)

(51)Int.Cl.[7]	F I		テーマコード（参考）
G06F 11/00	G06F 9/06	660N	5B076
G06F 12/00	G06F 12/00	520E	5B082

審査請求 有　請求項の数 18　OL　（全 11 頁）

(21)出願番号　特願2002-364550(P2002-364550)
(22)出願日　平成14年12月17日(2002.12.17)

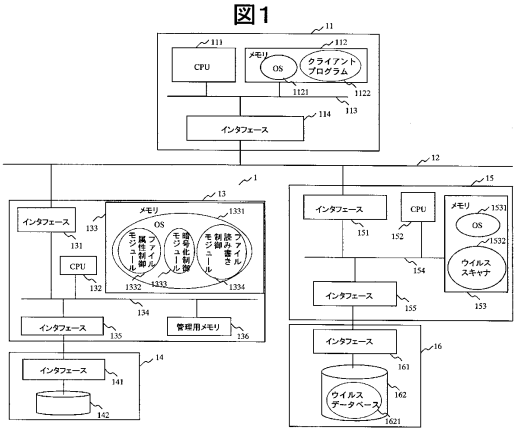
(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(74)代理人　100075096
弁理士　作田　康夫
(72)発明者　宮田　賢一
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社日立製作所システム開発研究所内
(72)発明者　堀内　孝
神奈川県横浜市戸塚区戸塚町５０３０番地
株式会社日立製作所ソフトウエア事業部内

Fターム(参考)　5B076 FD08
5B082 EA10

(54)【発明の名称】情報処理システム

(57)【要約】
【課題】ホスト計算機と記憶装置システムが接続されたファイルサーバシステムにおいて、不要なスキャン処理を削減してファイルアクセスを高速化するとともに、不正な情報の書き換えを防止して、より安全な情報処理システムを実現することにある。
【解決手段】ホスト計算機13上のCPU132が管理するファイルシステムに最終スキャン日時と最終更新者の各情報を暗号化して付加し、その情報に応じてホスト１３とスキャンサーバ15が連携して動作する。最終スキャン日時よりも古いファイルはスキャンを省略することでファイルアクセスを高速化する。アクセス対象のファイルがウイルスに感染していた場合はその原因となったユーザを特定し予め決めておいた対策を行う。
【選択図】図１

図1



【特許請求の範囲】
【請求項１】
ファイルの読み書きを行う計算機であって、
前記計算機は、前記ファイルの属性を表すファイル属性情報を備え、
前記ファイル属性情報を用いて、ファイルの安全性を確保するための処理を行う、計算機。
【請求項２】
ファイルの読み書きを行う計算機であって、
前記計算機は、前記ファイルの属性を表すファイル属性情報を備え、
前記ファイルに対するアクセスの要求があった場合、前記ファイル属性情報を用いて、ファイルの安全性を確保するための処理を行う、計算機。
【請求項３】
前記計算機は、ウイルスのパターンを管理するデータベースを更に備え、
前記データベースと前記ファイル属性情報を用いて、前記ファイルに対するウイルスを発見するための処理を行うか否かを判断する、請求項１記載の計算機。
【請求項４】
前記ファイル属性情報は、前記ファイルが最後にウイルスを発見するための処理が行われた日時を示す最終スキャン日時を含み、
前記計算機は、前記最終スキャン日時を用いて、ウイルスを発見するための処理が不要な場合は、当該処理を省略する、請求項３記載の計算機。
【請求項５】
前記計算機は、前記ファイルが更新された場合には、前記最終スキャン日時を所定の値に更新する、請求項４記載の計算機。
【請求項６】
前記データベースは、前記ウイルスのパターンが最後に更新された日時を示す最終更新日時を含み、
前記計算機は、前記最終更新日時と前記最終スキャン日時を比較し、
前記最終スキャン日時が前記最終更新日時より遅い場合は、前記ファイルに対するウイルスを発見するための処理を省略する、請求項４又は５記載の計算機。
【請求項７】
前記ファイル属性情報は、前記ファイルの内容を更新したユーザを表す最終更新者を含み、
前記計算機は、前記最終更新者を用いて、前記最終更新者に該当するユーザに対して、安全対策の処理を行う、請求項２記載の計算機。
【請求項８】
前記計算機は、前記ファイル属性情報を暗号化する、請求項１乃至７の１に記載の計算機。
【請求項９】
前記計算機は、装置の製造番号を暗号化のキーとする、請求項８記載の計算機。
【請求項１０】
ファイルの読み書きを行うホスト計算機と、前記ファイルに対してウイルスを発見するための操作を行うサーバ計算機を備えた情報処理システムにおいて、
前記ホスト計算機は、前記ファイルの属性を表すファイル属性情報を備え、
前記ファイル属性情報を用いて、ファイルの安全性を確保するための処理を行う、情報処理システム。
【請求項１１】
ファイルの読み書きを行うホスト計算機と、前記ファイルに対してウイルスを発見するための操作を行うサーバ計算機を備えた情報処理システムにおいて、
前記ホスト計算機は、前記ファイルの属性を表すファイル属性情報を備え、
前記ファイルに対するアクセスの要求があった場合、前記ファイル属性情報を用いて、フ

ァイルの安全性を確保するための処理を行う、情報処理システム。
【請求項１２】
前記サーバ計算機は、ウイルスのパターンを管理するデータベースを更に備え、
前記ホスト計算機は、前記データベースと前記ファイル属性情報を用いて、前記ファイルに対するウイルスを発見するための処理を行うか否かを判断する、請求項１０記載の情報処理システム。
【請求項１３】
前記ファイル属性情報は、前記ファイルが最後にウイルスを発見するための処理が行われた日時を示す最終スキャン日時を含み、
前記ホスト計算機は、前記最終スキャン日時を用いて、ウイルスを発見するための処理が不要な場合は、当該処理を省略する、請求項１２記載の情報処理システム。
【請求項１４】
前記計算機は、前記ファイルが更新された場合には、前記最終スキャン日時を所定の値に更新する、請求項１３記載の計算機。
【請求項１５】
前記データベースは、前記ウイルスのパターンが最後に更新された日時を示す最終更新日時を含み、
前記ホスト計算機は、前記最終更新日時と前記最終スキャン日時を比較し、
前記最終スキャン日時が前記最終更新日時より遅い場合は、前記ファイルに対するウイルスを発見するための処理を省略する、請求項１３又は１４記載の情報処理システム。
【請求項１６】
前記ファイル属性情報は、前記ファイルの内容を最後に更新したユーザを表す最終更新者を含み、
前記ホスト計算機は、前記最終更新者を用いて、前記最終更新者に該当するユーザに対して、安全対策の処理を行う、請求項１１記載の情報処理システム。
【請求項１７】
前記ホスト計算機は、前記ファイル属性情報を暗号化する、請求項１０乃至１６の１に記載の情報処理システム。
【請求項１８】
前記ホスト計算機は、装置の製造番号を暗号化のキーとする、請求項１７記載の情報処理システム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、ファイルシステムにおいて、ウイルス感染を検索する技術に関する。
【０００２】
【従来の技術】
近年、情報量の増加に伴い、記憶装置システムの大規模化が進んでいる。ファイルを提供するサーバ（以下、ファイルサーバと略す）を同時に多数のユーザで共有する場合、ファイルに対するアクセス（以下、ファイルアクセスと略す）の速度が低下するとともに、アクセスするデータの安全性に問題が生じる。即ち、ファイルサーバ内のファイルがコンピュータウイルス(以下、ウイルスと略す)に感染すると、多くのユーザに影響を及ぼす。
【０００３】
そこでウイルス感染の被害を食い止めるために、ウイルスを発見するためのソフトウェア（以下、ウイルススキャナと呼ぶ）を記憶装置システムに組み込み、ファイルアクセスの度に、ファイルにウイルスが感染しているか否かの検査が行われる。ここで、大規模なファイルサーバでは、ウイルススキャナの性能を向上させることが求められる。
【０００４】
ウイルススキャナの性能を向上させるための従来技術は、ウイルス発見のための処理（以下、スキャンと呼ぶ）の適用要否を決定するために、スキャン済みのファイル一覧を検索

し、見つかればスキャンを省略する、というものである（例えば、非特許文献１参照）。
【０００５】
【非特許文献１】
John Phillips, “Antivirus Scanning Best Practices Guide”,[online],2002年5月28日,NetApp Technical Library,[2002年9月24日検索],インターネット<URL:http://www.netapp.com/tech_library/3107.html>
【０００６】
【発明が解決しようとする課題】
上述の方法によると、スキャン済みのファイル一覧を、スキャンの要求が発行される度に走査するため、ファイル一覧のファイル数の増大により、走査のためのコストが増大し、スキャンの高速性が得られない、という問題がある。
【０００７】
また、スキャン済みのファイル一覧に不正にファイルを追加することにより、本来スキャンが必要なファイルを不要であるように見せかけることができ、アクセスに対する記録（以下、ログと呼ぶ）の採り方によっては、不正なアクセスにより、ログそのものも改竄されるおそれがある、という問題がある。
【０００８】
更に、ファイルにウイルスが感染した原因となったユーザを特定し適切な対策をとるためには、膨大な量のファイルアクセス記録やユーザのサーバログイン記録を検査しなければならない、という問題がある。
【０００９】
本発明の目的は、不要なスキャンの処理を削減して、ファイルアクセスを高速化すること、及び、不正な情報の書き換えを防止して、より安全な情報処理システム実現することにある。
【００１０】
【課題を解決するための手段】
上記目的を解決するために、計算機は、ウイルスの様々なパターンを管理するデータベースであるウイルスデータベースを備え、また、ファイル属性情報として、ファイルがスキャンされた最後の日時を表す最終スキャン日時を備える。ウイルスデータベースは、当該ウイルスデータベースが最後に更新された最終更新日時を記録しており、最終スキャン日時と最終更新日時を比較することによって、不要なスキャンの処理を削減する。また、ファイル属性情報として、ファイルに対して最後のスキャンを行ったユーザを表す最終更新者を付加する。最終スキャン日時と最終更新者を暗号化して保存することにより、不正な書き換えを防止し、情報処理システムの安全性を高める。
【００１１】
【発明の実施の形態】
以下、本発明の実施形態を詳細に説明する。
【００１２】
図1は、本発明に係る、ファイルサーバを含むシステム（以下、ファイルサーバシステムと呼ぶ）を構成要素とする、情報処理システムの例を示す。尚、本図において、ソフトウェアは丸で囲んでいる。
【００１３】
ファイルサーバシステム１は、ネットワーク12を介して、クライアント計算機（以下、クライアントと略す）11と、後述のスキャンサーバ15に接続されている。図では簡単のために１台のクライアントを記しているが、複数台でもよい。
【００１４】
ファイルサーバシステム１は、ホスト計算機（以下、ホストと略す）13、記憶装置システム14とからなる。
【００１５】
ホスト13は、ネットワーク12と接続するためのインタフェース131、CPU132、オペレーテ

ィングシステム（以下、OSと略す）1331を内蔵するメモリ133、ファイルサーバ内ネットワーク134、記憶装置システム14と接続するためのインタフェース135、装置に関する情報を格納する管理用メモリ136とから構成される。CPU132は、メモリ133に内蔵されるOS1331を読み出して、処理を行う。

【００１６】

OS1331は、ファイル属性制御モジュール1332、暗号化制御モジュール1333、ファイル読み書き制御モジュール1334等のプログラムを有する。

【００１７】

ファイル属性制御モジュール1332は、ファイル属性（図2）の読み書きを行うためのプログラムである。

【００１８】

暗号化制御モジュール1333は、ファイル属性の暗号化と復号を行うためのプログラムである。

【００１９】

ファイル読み書き制御モジュール1334は、ファイルの実データの読み書きを行うためのプログラムである。

【００２０】

管理用メモリ136は、装置の製造番号の情報を格納しており、OS1331を実行するCPU132からの要求によりCPU132に製造番号を渡す。

【００２１】

記憶装置システム14は、ホスト13と接続するためのインタフェース141と記憶装置142とからなる。一般に、記憶装置142は磁気媒体を使うことが多いが、光学的媒体などの他の媒体を利用する装置であってもよい。

【００２２】

スキャンサーバ15は、ネットワーク12と接続するためのインタフェース151、CPU152、OS1531とウイルススキャナ1532を内蔵するメモリ153、スキャンサーバ内ネットワーク154、記憶装置システム16と接続するためのインタフェース155とからなる。

【００２３】

記憶装置システム16は、スキャンサーバ15と接続するためのインタフェース161、ウイルスデータベース1621を格納する記憶装置162とからなる。

【００２４】

ウイルスデータベース1621は、ウイルスの様々なパターンを管理するデータベースであり、一定期間毎に、または、不定期に、更新される。また、ウイルスのパターンが更新された最後の日時を示す最終更新日時を記録している。

【００２５】

CPU152は、ウイルススキャナ1532を実行することにより、対象のファイルに係るウイルスデータベース1621に含まれる各パターンと比較しながらスキャンを行う。

【００２６】

クライアント11は、CPU111、OS1121とクライアントプログラム1122を内蔵するメモリ112、クライアント内ネットワーク113、ネットワーク12と接続するためのインタフェース114とからなる。CPU111は、OS1121を実行することにより、クライアント１１の指示をホスト13に伝える。

【００２７】

尚、図では、メモリ133と管理用メモリ136を別々に記しているが、同じメモリであってもよい。また、スキャンサーバ15をホスト13と異なる計算機として記述しているが、スキャンサーバ15の機能をホスト13に内蔵してもよい。

【００２８】

以下、CPU132がOS1331を実行することによりファイルを読み書きするための、一般的な処理について説明する。

【００２９】

まず、CPU132は、ファイルにアクセスするための準備をする（以下、この操作を、ファイルをオープンする、と呼ぶ）。次に、オープンしたファイルに対して読み書きを行う。そして、そのファイルを使用しなくなった時点で、後始末を行う（以下、この操作を、ファイルをクローズする、と呼ぶ）。

【００３０】

この一連の流れは、ネットワーク12を介して、CPU111がクライアントプログラム1122を実行することにより行うことができる。すなわち、クライアント１１に対してファイルオープンの要求が発行されると、CPU111はファイルサーバシステム1に対して該要求を通知し、ホスト13上で処理が行われる。ファイル操作の要求についても、同様である。

【００３１】

次に、ファイルをオープンする時の一般的な処理について説明する。

【００３２】

まず、CPU132は、クライアント11からのファイルオープンの要求を受けると、ファイル(以下、このファイルをFとする)のスキャン要求とFをスキャンサーバ15に送信する。そして、スキャンサーバ15は、Fに対するスキャンを行う。

【００３３】

次に、スキャンサーバ15からスキャンの結果を受信し、該結果をもとに、Fがウイルスに感染しているか否かを判定する。

【００３４】

感染していない場合、Fをオープンし、Fのオープンが成功したことをクライアント11に通知する。感染している場合、Fのオープンが不可能であることをクライアント11に通知する。その後、ファイルの読み書き処理へ移行する。

【００３５】

図2は、本発明に係る、ファイル属性の構成例である。

【００３６】

ファイル属性は、更新日201、所有者202、参照権203、最終スキャン日時204、最終更新者205、データブロック206とからなり、記憶装置142に格納される。

【００３７】

更新日201は、ファイルの内容を最後に更新した日時を示す。

【００３８】

所有者202は、CPU132が管理するユーザを一意に識別するための番号を示す。

【００３９】

参照権203は、そのファイルを読み書きできるか、または、実行できるか、などの権利を示す。

【００４０】

最終スキャン日時204は、そのファイルがスキャンされた最後の日時を表し、スキャン実行時に更新される。

【００４１】

最終更新者205は、最後にそのファイルの内容を更新したユーザを表し、ファイルへのデータ書き込み時に更新される。

【００４２】

データブロック206は、記憶装置142上のデータの位置を表すブロック番号の配列である。

【００４３】

CPU132は、ファイルの新規作成時に、ファイル属性制御モジュール1332を実行することにより、図2のファイル属性の各値を初期設定する。また、OS1331上のコマンドを用いて明示的に各値を変更する。ただし最終スキャン日時204と最終更新者205は明示的に変更できない。

【００４４】

尚、最終更新者として、最後にファイルの内容を更新した者のみならず、一定回数分の履歴を保存しておくことも可能である。１つは、１ファイル当たりの平均更新間隔をウイル

スパターンの平均更新間隔で割った値の分だけ更新者を記録する方法がある。例えば、一定期間のファイルに対するアクセスの統計を取ってみて、１日に平均１回ファイルが更新されることが分かり、ウイルスパターンが５日に１回更新されるならば、過去５回分の更新者を記録する。または、スキャンした後、ウイルスデータベースが更新されるまでのすべての更新者を記録しておく方法もある。

【００４５】

図2の構成は、従来の構成に対して、最終スキャン日時204と最終更新者205が加わった構成となっている。これらの値によって、スキャンの必要性を決定し、ウイルス感染の原因となったユーザを特定することができる。最終スキャン日時204と最終更新者205は、不正に侵入したユーザが不正にそれぞれの欄を書き換えることを防止するために、両者ともに暗号化して保存されるのが好ましい。

【００４６】

尚、最終スキャン日時204と最終更新者205は、お互いに依存するものではなく、どちらか一方のみをファイル属性の構成としてもよい。また、参照権についての詳細は、「The Design and Implementation of the 4.3BSD UNIX（登録商標）Operating System」(Samuel J. Leffler, et al, Addison-Wesley, 1989)第58ページから第60ページに、データブロックの管理方法についての詳細は、第191ページから第195ページにあるのでここでは省略する。

【００４７】

図3は、本発明に係る、ファイルオープンの処理フローである。尚、以下の暗号化と復号では、暗号化アルゴリズムの一つであるDESアルゴリズムを用いる。

【００４８】

まず、CPU132は、暗号化制御モジュール1333を実行することにより、Fに対する最終スキャン日時（以下、この日時をXとする）を復号する(ステップ301)。次に、復号が正常に行えたか否かを判定し（ステップ302）、正常ならば、スキャンサーバ15にウイルスデータベース1621の最終更新日時（以下、この更新日時をYとする）を問い合わせる(ステップ303)。正常でなければ、予めメモリ133に登録しておいた管理者に対して、電子メールなどにより、その旨を通知する(ステップ304)。

【００４９】

スキャンサーバ15は、Yについての問い合わせを受信し、CPU152は、記憶装置162からYを読み出し、ホスト13に対して応答する（ステップ305）。そして、ホスト13は、応答を受けて（ステップ306）、CPU132は、XとYを比較する（ステップ307）。

【００５０】

Xの方が早い(X<Y)場合、つまりウイルスデータベース1621が更新される前にFのスキャンが行われている場合は、最新のウイルスに感染している可能性があるためFのスキャンを行い(ステップ308)（詳細は図４にて後述）、スキャンの結果をホスト13に通知する（ステップ309）。Xの方が遅い場合は、スキャンの必要がないため、CPU132は、Fをオープンする（ステップ312）。

【００５１】

スキャンサーバ15からステップ308の結果を受信すると、CPU132は、該結果をもとに、ウイルスに感染しているか否かを判定する(ステップ310)。感染していなければ、CPU132は、暗号化制御モジュール1333を実行することにより、現在時刻を、装置の製造番号Kをキーとして暗号化し、最終スキャン日時404を更新する(ステップ311)。その後、CPU132は、Fをオープンし(ステップ312)、オープン成功をクライアント11に通知する(ステップ313)。

【００５２】

ステップ310でFがウイルスに感染していると判定された場合、ウイルス感染の原因となったユーザを特定するために、CPU132は、ファイルの最終更新者205に対する処理を行う(ステップ314)。この処理については、後述する（図5）。その後、CPU132は、オープン不能をクライアント11に通知し(ステップ315)、処理を終了する。

【００５３】
ステップ313の後、CPU132は、Fに対するアクセス要求が読み出し要求か否かを判定し、読み出し要求の場合、Fの読み出しを行い（ステップ317）、Fをクローズし（ステップ321）、処理を終了する。
【００５４】
書き込み要求の場合、CPU132は、Fに対する書き込みを行う（ステップ318）。次に、Fの書き込みをするユーザの識別番号Uを、装置の製造番号Kをキーとして暗号化し、最終更新者205を更新する（ステップ319）。その後、ファイル属性を更新する（ステップ320）。ここで、ステップ311にて更新した最終スキャン日時204を、更に0に更新する。そうすれば、次回、ファイルにアクセスされた場合、ステップ307で、必ずX<Yとなり、ステップ308に進むことになる。これは、ファイルが更新された時にウイルスに感染する可能性が高いため、次回のアクセス時には、スキャンを行う必要があるからである。
【００５５】
尚、最終スキャン日時204を用いない場合は、ステップ301～307、311の処理を省略する。この時、ホスト13は、スキャン要求を送信し、スキャンサーバ15は、該要求を受信してから、ステップ308の処理に移行する。また最終更新者205を用いない場合は、ステップ314を省略する。
【００５６】
また、ウイルススキャンを行うタイミングは、ファイルのオープン時としたが、ファイルのオープン時とクローズ時の両方で行ってもよい。デフォルトではオープン時のみ、ユーザの指示によりクローズ時にもスキャンを行う、としてもよい。
【００５７】
次に、図３のステップ308の詳細について、図４を用いて説明する。
【００５８】
まず、CPU152は、ホスト13からスキャン要求と、スキャンするFを受け取り(ステップ401)、スキャンするFの実データを読み出す(ステップ402)。
【００５９】
その後、ウイルスデータベース1621からウイルスのパターンを一つ読み出す(ステップ403)。ここで、読み出した１つのパターンを、以下、Pとする。Pにあてはまるウイルスが Fに感染しているか否かを検査する(ステップ404)。
【００６０】
感染していない場合、まだ未検査のパターンがウイルスデータベース1621上にあるかを判定し(ステップ405)、あればステップ403の処理に戻る。未検査のパターンがなければ、Fはウイルスに感染していないと判定し、ホスト13に、非感染であることを通知する(ステップ406)。感染している場合、ホスト13に、感染していることを通知する(ステップ407)。
【００６１】
尚、ステップ308において、最終スキャン日時204をスキャンサーバ15に渡すためのインタフェースをスキャンサーバ15に設け、XとYの比較を、ステップ404の前に行う（パターン毎にスキャン要否を判定する）ことにより、該処理の回数を削減することも可能である。
【００６２】
図5は、ステップ314の、CPU132が暗号化制御モジュール1333を実行することにより行う処理フローである。
【００６３】
まず、CPU132は、Fの最終更新者205を復号し、これをUとする(ステップ501)。ここで、復号の手順の一例を以下に示す。
【００６４】
復号のための入力をEとして、暗号化されている値を与える。そして、復号のためのキーをKとして、暗号化するときに用いたものと同じ装置の製造番号を与える。ここで、製造番号をキーとするのは、クライアントにとって、知ることが困難なデータだからである。

更に、Kをキーとして Eを、復号する（この結果をDとする）。最後に、Dを復号の要求元に返却する。

【００６５】

ステップ501の後、CPU132は、復号が正常に行われたか否かを判定し(ステップ502)、正常ならば、Uに対する安全対策を行う(ステップ503)。安全対策の一例については後述する（図6、図7）。正常でなければ、メモリ133に予め登録しておいた管理者に対して、電子メールなどにより、その旨を通知する(ステップ504)。

【００６６】

尚、DESアルゴリズムの詳細については「コンピュータセキュリティの基礎」(Deborah Russell他著、山口英訳、アスキー出版局1997)第213ページから第227ページにあるのでここでは省略する。また、暗号化アルゴリズムとしては、暗号化したデータを復号することのできる可逆暗号化アルゴリズムであれば、DES以外のアルゴリズムを用いても良い。更に、暗号化のキーとしては、暗号化と復号で常に同一の値となることが保証できれば、製造番号以外でもよい。

【００６７】

図6は、メモリ133が有するウイルス感染原因者対策表の例である。

【００６８】

ステップ501でファイルの最終更新者が得られた場合、CPU132は、図6の表を用いて該ユーザに対する処理を決定する。表はファイル名601と、601に対応する処理とからなる。本図では、処理の例として、該ユーザに警告を通知する処理602、ホスト13の管理者に通知する処理603、該ユーザがホスト13にアクセスすることを禁止する処理604を挙げる。ファイル601に対して、602から604の処理のいずれか、または、その組み合わせをホスト13の管理者などが決定し、表に記入しておく。

【００６９】

処理604が設定されている場合、該ユーザがホスト13上にアクセスできないように、例えば、メモリ133内にホスト13にアクセスすることを禁止すべきユーザのリストを用意し、そのリストに該ユーザを登録することで、以降の該ユーザのアクセスを拒否することができる。

【００７０】

特定のファイルに対する処理のほか、それ以外のすべてのファイル（特定する必要がないファイル）に対する処理も記述しておく。この記述により図6の表にCPU132が管理するすべてのファイルを記述する必要がなくなり、記述すべき情報量の増大を抑えることができる。

【００７１】

図7は、ウイルス感染原因者対策の、CPU132が行う処理フローである。

【００７２】

図6の表から、Fに対するエントリを探す(ステップ701)。次に、表中にFに対するエントリがあるかを判定する(ステップ702)。エントリがある場合、該エントリに記述されている処理を実行する(ステップ703)。一方、エントリがない場合、「その他すべて」のエントリを探し、そのエントリに記述されている処理を行う(ステップ704)。

【００７３】

【発明の効果】

本発明によれば、不要なスキャン処理を削減して、ファイルアクセスを高速化できる。また、不正な情報の書き換えを防止して、より安全な情報処理システムを実現できる。

【図面の簡単な説明】

【図１】情報処理システムの図である。

【図２】ファイルの構成例である。

【図３】スキャン処理の処理フローである。

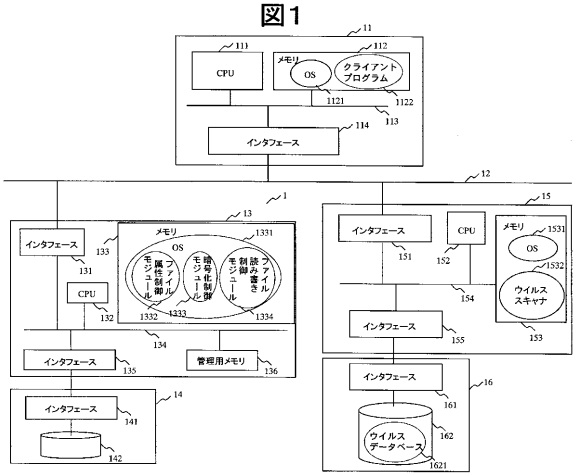
【図４】ファイルオープンの処理フローである。

【図５】最終更新者確認の処理フローである。

【図６】ウイルス感染原因者対策表の例である。
【図７】ウイルス感染原因者対策の処理フローである。
【符号の説明】
1…ファイルサーバシステム、11…クライアント計算機、12…ネットワーク、13…ホスト計算機、14…記憶装置システム、131…インタフェース、132…CPU、133…メモリ、1331…OS、142…記憶装置、15…スキャンサーバ、153…ウイルススキャナ

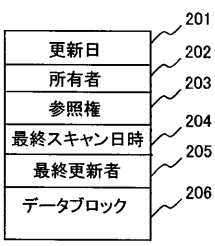
【図１】

図1



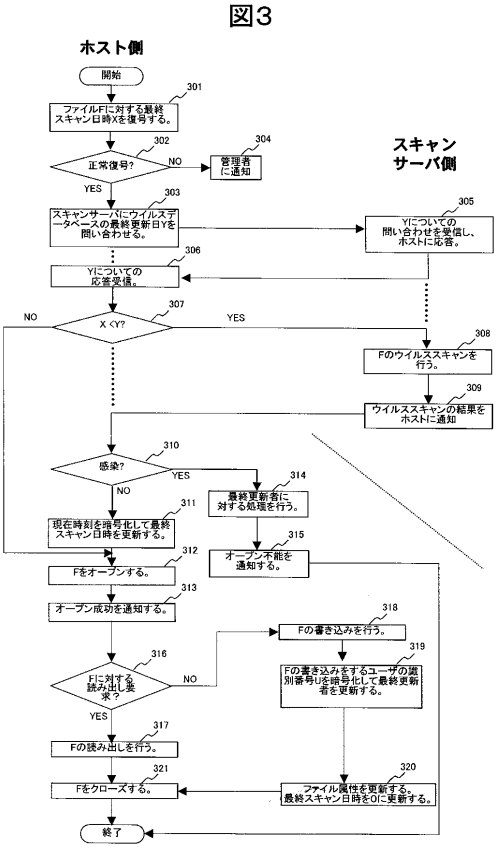
【図２】

図2



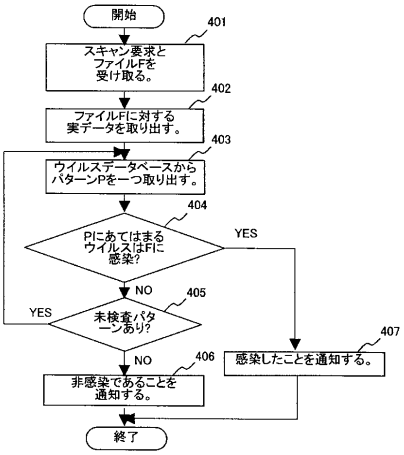
【図３】

図3



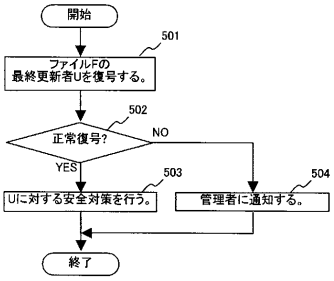
【図４】

図4



【図５】

図5



【図６】

図6

601 ファイル	602 当該ユーザに警告通知	603 ホスト管理者に通知	604 当該ユーザのホストアクセス禁止
file1.doc	√		
file2.doc		√	
その他すべて		√	√

【図７】

図7