(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4870937号
(P4870937)
(45)発行日　平成24年2月8日(2012.2.8)　(24)登録日　平成23年11月25日(2011.11.25)

(51)Int.Cl.　F I
G06F 21/22 (2006.01)　G06F 9/06 660G

請求項の数 8 (全 15 頁)

(21)出願番号	特願2005-92511(P2005-92511)
(22)出願日	平成17年3月28日(2005.3.28)
(65)公開番号	特開2005-316974(P2005-316974A)
(43)公開日	平成17年11月10日(2005.11.10)
審査請求日	平成20年3月24日(2008.3.24)
(31)優先権主張番号	10/836,409
(32)優先日	平成16年4月30日(2004.4.30)
(33)優先権主張国	米国(US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(72)発明者　カグラー　ガナャクティ
アメリカ合衆国　９８０５２　ワシントン州　レッドモンド　ワン　マイクロソフト　ウェイ　マイクロソフト　コーポレーション内

最終頁に続く

(54)【発明の名称】ソフトウエアのアップデートを制限する方法およびシステム

(57)【特許請求の範囲】
【請求項１】
サーバコンピュータにおいて海賊版ソフトウェアを検出する方法であって、
クライアントコンピュータから、クライアントコンピュータに常駐しているソフトウェアプログラムへのソフトウェアアップデートに対する要求を受信するステップと、
前記サーバコンピュータにおいて、前記クライアントコンピュータ内の前記ソフトウェアプログラムに対して実施すべき、実行可能コードを含むテストを決定するステップと、
前記テストを前記クライアントコンピュータへ送信するステップと、
前記クライアントコンピュータにおいて、前記テストの前記コードを実行するステップと、
評価のために前記テストの結果を前記サーバコンピュータへ送信するステップであって、前記テストの前記結果は前記サーバコンピュータによって使用されて前記クライアントコンピュータにおける前記ソフトウェアプログラムが適法なコピーであるかどうかを決定するステップと、
前記クライアントコンピュータが、前記サーバコンピュータによって評価されたように前記テストに合格せずに前記テストが海賊版ソフトウェアを検出する場合、前記クライアントコンピュータに前記ソフトウェアアップデートを拒否するステップと
を備えることを特徴とする方法。
【請求項２】
ソフトウェアアップデートに対する要求を受信する前記ステップは、前記サーバコンピ

ュータが、ネットワークを介して前記ソフトウェアアップデート要求を受信することを含み、前記サーバコンピュータは、前記ソフトウェアプログラムが適法なコピーであるかどうかを決定するときに使用される海賊行為検出ソフトウェアをホスティングすることを特徴とする請求項１に記載の方法。

【請求項３】

前記テストを実行する前記ステップは、クライアントソフトウェアプログラムの実行時の完全性検査を実施することを含むことを特徴とする請求項１に記載の方法。

【請求項４】

前記実行時の完全性検査は、クライアントソフトウェア内に存在する実行可能なプログラムを実行すること、巡回冗長符号検査を実施すること、ハッシュ値検査を実施すること、およびデジタル署名を検証することのうち１つまたは複数を含むことを特徴とする請求項３に記載の方法。

【請求項５】

クライアントコンピュータから、ソフトウェアプログラムアップデートに対する要求を受信するステップと、

前記クライアントコンピュータ内の前記ソフトウェアプログラムに対して実施すべき、実行可能なテストコードを含むテストを決定するステップと、

前記サーバコンピュータから前記クライアントコンピュータへ、前記クライアントコンピュータ内で実行する前記テストコードをダウンロードするステップと、

前記クライアントコンピュータにおいて実行された前記テストの結果を受信するステップであって、前記テストの前記結果は前記サーバコンピュータによって使用されて前記クライアントコンピュータ内の前記ソフトウェアプログラムが適法なコピーであるかどうかを決定するステップと、

前記テストの前記結果を評価することによって、前記クライアントコンピュータソフトウェアプログラムが改変されたかどうかを決定するステップと、

前記テストの前記結果の評価に依存して前記ソフトウェアアップデートを許可するかどうかを決定するステップと

を備える方法をサーバコンピュータ上で実行するためのコンピュータ実行可能命令を有するコンピュータ読取り可能記憶媒体。

【請求項６】

前記方法は、前記ソフトウェアプログラムが改変されたことを前記テスト結果が示している場合に、前記ソフトウェアアップデートを拒否するステップをさらに含むことを特徴とする請求項５に記載のコンピュータ読取り可能記憶媒体。

【請求項７】

海賊版ソフトウェアを検出するシステムであって、

サーバコンピュータおよびクライアントコンピュータの間でメッセージを送信するネットワークと、

前記ネットワーク上に常駐し、クライアントコンピュータ用のソフトウェアアップデートを有するサーバコンピュータと、

前記ネットワーク上に常駐し、前記サーバコンピュータから、クライアントソフトウェアのアップデートを要求するクライアントコンピュータとを備え、

前記クライアントソフトウェアに対して前記クライアントコンピュータによってテストが実施され、前記テストは、前記サーバコンピュータから前記クライアントコンピュータにダウンロードされ、前記クライアントコンピュータ上で実行できる実行可能ソフトウェアプログラムテストコードを含み、前記テストは前記ソフトウェアプログラムテストコードの実行結果が、前記クライアントソフトウェアが海賊版であるかどうかを決定するために前記サーバコンピュータに送られ、前記テスト結果が前記サーバコンピュータ内で合格しなければ、前記クライアントコンピュータはクライアントソフトウェアの前記アップデートを拒否されることを特徴とするシステム。

【請求項８】

前記テストは、クライアントソフトウェアプログラムの実行時の完全性検査を実施するように設計されていることを特徴とする請求項７に記載のシステム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、一般に、コンピュータソフトウエアのセキュリティの分野に関する。より詳細には、本発明は、海賊版ソフトウエアまたはライセンスを受けていないソフトウエアの検出に関する。

【背景技術】

【０００２】

ソフトウエアの海賊行為は、ソフトウエア製造業者の重大な懸念である。金銭収入が大きく減っているのは、市販されているソフトウエア製品の不正コピーおよび不正使用に起因すると考えられる。ソフトウエア製品は、数多くの形態で保護することができる。普及している１つの形態は、ソフトウエア製品内に埋め込まれたコードを使用することである。このようにコードを埋め込むことによって、不正にコピーされたコードは一般に、動作させることが難しくなる。これらのタイプのソフトウエアベースのコードによる保護は、熟練していないソフトウエア盗用者の活動を阻止するには有用であるが、熟練したソフトウエアハッカーにとっては、このような保護では、たかだかその活動のペースを落とさせるだけである。実際、ソフトウエアハッカーは、コードを解析・模倣してパッチその他の回避方法を考案し、それによってソフトウエアベースの保護に打ち勝つことができる助けとなるいくつかの市販のツールが、インターネット上で入手可能である。したがって、現在利用可能なソフトウエアベースのコードによる保護によっては、市販ソフトウエアの不正使用を防止する効果はますます薄くなりつつある。しかし、ソフトウエアベースのコードによる保護は、ある程度のレベルで海賊行為から確かに保護する働きをしており、ある種のソフトウエア海賊行為予備軍に対しては効果的である。

【発明の開示】

【発明が解決しようとする課題】

【０００３】

そのため、ソフトウエアコードによる保護の概念のてこ入れをし、コードを解析・模倣しこれらの保護を阻害する頭のよい盗用者の問題に対処するための、盗用者からソフトウエアを保護する技術が必要とされている。本発明は、本明細書に記載するさらなる利点によって、上記要求に対処し、それらの問題を解決する

【課題を解決するための手段】

【０００４】

海賊版ソフトウエアを検出する方法は、クライアントコンピュータからのソフトウエアアップデート要求を受け取るステップと、このクライアントコンピュータに実施すべきテストを提供するステップとを含む。このテストは、クライアントコンピュータ上で、クライアントソフトウエアアプリケーションに対して実施する。このテストの結果、クライアントコンピュータは、ソフトウエアのアップデートを拒否されることがあり、それによって、ソフトウエアの盗用が広がるのを阻止することができる。

【０００５】

本発明の一態様によれば、クライアントは、サーバにソフトウエアのアップデート要求を出し、サーバはクライアントにテストコードを送信し、クライアントはこのコードを実行し、その結果をサーバに送出する。サーバは、この結果を評価し、このテストによりサーバがクライアントのソフトウエアが本物であるかどうかを検証することに基づいて、ソフトウエアのアップデート要求を許可または拒否する。サーバを使用しない環境では、アップグレードを要求するクライアントコンピュータにＣＤからテストを提供することができる。アップグレード要求の許可または拒否は、クライアントコンピュータによってソフトウエアアプリケーション上で実施される完全性テストの合格または不合格にそれぞれ基づいて行われる。

【０００６】
上記概要ならびに好ましい実施形態の以下の詳細な説明は、添付の図面を併せ読むことによりよく理解されるだろう。本発明の実施形態を例示するために、これらの図面に本発明の構成の例を示す。ただし、本発明は、ここで開示する特定の方法および手段に限定されるものではない。
【発明を実施するための最良の形態】
【０００７】
概要
アプリケーションに付加された海賊行為防止コードを用いることによって、ソフトウエアベースの海賊行為防止技術を有利に利用して、使用中のソフトウエアが本物のライセンスを受けたものであるかどうかを識別する助けとすることができる。ソフトウエアの盗用者は、ソフトウエアベースの保護を改変し無効にしてこれらの保護方式を欺き、それによってこのアプリケーションを使用可能にする。
【０００８】
本発明の一つの態様は、コードの完全性が疑わしいことを示す結果が戻されるように、アプリケーション上でこの疑わしいコードに対してテストが実施される場合に、このような海賊版アプリケーションに対する正当なアップデートを拒否できることである。ソフトウエアベースの海賊行為保護などのアイテムが適切に存在し、かつ／または適切に機能していることをテストすることによって、存続可能なライセンスの正当性に関する評価を行うことができる。疑わしいアプリケーションをホスティングするクライアントコンピュータによって実施されたテストが不合格になった場合、このコードは改変されたものである可能性があり、まず間違いなく、正しいライセンスを受けたものではない。このような状態の下では、ソフトウエアのアップデートを拒否することができ、それによって、海賊版コードをアップグレードし続けることを妨げることができる。先々のアップグレードを妨げることにより、最終的には海賊版ソフトウエアが旧版になり、したがって、ソフトウエアの盗用版の有用性は少なくなる。
【実施例１】
【０００９】
消費者に販売されるソフトウエアアプリケーションに設けられるソフトウエアベースの保護に対する攻撃には、３つの主なカテゴリーがある。第１のカテゴリーは、データ攻撃と呼ぶことができるものである。この攻撃では、制限を超える状態を感知することの基礎になるデータを変更することによって、ソフトウエアによる保護の仕組みの裏をかく。例えば、試用版など、ユーザのマシンにインストールされるソフトウエアアプリケーションが、１週間または１ヵ月間有効である場合、ソフトウエアによる保護の仕組みにより、１週間または１ヶ月の有効期限が過ぎると、このアプリケーションにアクセスはできなくなる。この保護の裏をかく一方法は、ソフトウエア保護コードを入力し、タイマ情報（データ）を除去するか、または他の方法で改ざんして、期限切れに到達させないことである。このように、盗用者は、期間が制限された試用アプリケーションソフトウエアライセンスを取得し、このライセンス期間を不正に無限に延長する。よく用いられる別の技法は、試用ソフトウエアをブートするたびに、この試用ソフトウエアのタイマをゼロにリセットすることである。このように、仮ライセンス期間を不正に延長する。こうした種類の攻撃をデータ攻撃と呼ぶことがある。
【００１０】
ソフトウエアベースの盗用防止の仕組みに対する第２のカテゴリーの攻撃は、バイナリパッチ（ｂｉｒａｎｙ　ｐａｔｃｈ）と呼ぶことができるものである。このよく用いられる技法は、ソフトウエア盗用者が、海賊版アプリケーションが機能し続けるように、その中で働くコードを生成するというものである。なぜなら、コードパッチにより、ライセンステストパラメータが裏をかかれるからである。例えば、ライセンス版ソフトウエアが、所与の期間で期限切れになる場合、あるいは、他のなんらかのライセンス存続可能性テストが用いられる場合、盗用者は、ソフトウエアパッチを使用して、ライセンスが失効した

ときでも、テストの状態を常にライセンスされている状態に設定することができる。このタイプの攻撃をバイナリ攻撃と呼ぶことがある。

【００１１】

ソフトウエアベースの盗用防止保護に対する第３のタイプの攻撃は、バイパスキー（ｂｙｐａｓｓ　ｋｅｙ）の漏洩と呼ぶことができるものある。このタイプの攻撃では、通常、ソフトウエアアプリケーションのサイトライセンスとして企業に与えられるマルチユーザキーを改変し、それによって不正なキー所有者が、所望のソフトウエアアプリケーションのコピーを入手することができる。この一例は、このキーを正当に購入した企業の外部の人間に、このバイパスライセンスキーを販売して拡散させることであろう。この漏洩キーおよび製品識別子が他人に渡ると、それらを用いることで、所望のソフトウエアの動作版を入手することができる。

【００１２】

一般に、バイパスキー漏洩タイプの攻撃が発見された場合、このバイパスキーを変更することができ、古いキーを用いても、海賊版ソフトウエアへのアップグレードが拒否される。現在、このタイプの攻撃の解決策は、当技術分野では周知のものである。しかし、データタイプの攻撃およびバイナリタイプの攻撃を検出しそれに打ち勝つことは、それほど簡単ではない。

【００１３】

データタイプおよびバイナリタイプの攻撃は、盗用されたアプリケーションのソースコードに対してテストを実行することによって検出することができる。これらのテストにより、このソースコードが、製造業者によって最初に配給されたコードと同じであることが明らかになれば、テストされたアプリケーションプログラムは、存続可能とみなすことができる。しかし、アプリケーションコードが、製造業者によって配給されたものと異なる場合、このコードは改変されている可能性がある。コードの改変は合法であることもあるし、改変されたコードは、海賊版コードを示している改ざんを示すこともある。

【００１４】

製造業者の当初の製品と一致しないアプリケーションソフトウエアコードは、この製造業者から本物のアップデートを取得する通常のプロセスの結果であることがある。これらのアップデートは一般に、新しいバージョン、あるいはこのアプリケーションに適用されるサービスパックアップデートとして、このアプリケーションに反映される。そのため、アプリケーション上で示されるバージョンまたはソフトウエアサービスパックに応じて、ソフトウエアアプリケーションコードの様々なテストが必要になり得る。アプリケーションソフトウエアの成熟レベルに応じて、巡回冗長符号検査、チェックサム、またはハッシュ値などのテストを適用しなければならない。

【００１５】

アプリケーションソフトウエアの完全性を判定するために実施できる別のタイプのテストは、ソフトウエアアプリケーション内の隠しコードを実行することである。アプリケーションソフトウエア内に隠されたコードは、アプリケーションの実用性の点では遊んでいることがあるが、完全性テストが望まれる場合に活動化させることができる。別のソフトウエアベースの盗用防止手段では、アプリケーションソフトウエアに様々なデジタル署名を挿入することができる。デジタル署名が不成功になることは、盗用者に対してライセンス許可が延長されるようにアプリケーションコードが改変された可能性があることを示し得る。本発明の一態様においては、アプリケーションソフトウエアに対して、そのライセンスの完全性を判定するテストをオンラインで実施できる。

【００１６】

図１は、本発明の態様を実施できるネットワーク構成１００を示す。この構成では、ネットワーク２０は、クライアントコンピュータＡ３０、クライアントコンピュータＢ４０、およびクライアントコンピュータＣ５０をネットワークサーバ１０に相互接続する。１つのサーバおよび３つのクライアントしか示していないが、図１の構成は単なる例であり、サーバの数を増やすこともできるし、同様に、クライアントコンピュータの数を増減す

ることもできる。このネットワーク構成においては、サーバ１０は、要求された時に、クライアントコンピュータ３０、４０、および５０にソフトウエアのアップデートを提供できるサーバとして指定できる。

【００１７】

好ましい実施形態においては、クライアントコンピュータ３０、４０、または５０は、サーバ１０と交信して、クライアントコンピュータ上に常駐するソフトウエアアプリケーションのアップグレードを要求するのが好ましい。サーバ１０は、要求されたアップグレードを提供できる。本発明の一つの態様においては、サーバ１０は、クライアントソフトウエアアプリケーション用の完全性テストコードを提供することもできる、このコードは、クライアントマシンに転送し、そこで実行することができる。次いで、クライアントコンピュータＡ３０、Ｂ４０、およびＣ５０は、テスト結果をサーバ１０に送り返して評価を受ける。この評価結果により、クライアントが海賊版ソフトウエアを実行している可能性に基づいて、クライアントがソフトウエアアプリケーションアップグレードにアクセスすることを許可するか、あるいは、アップグレード要求を拒否するかを判定できる。

【００１８】

図２は、本発明の態様を組み込んだ方法の例である。少なくとも１つのクライアントおよび１つのサーバを含むシステムでは、クライアントコンピュータが、クライアントソフトウエアアプリケーションのアップデート要求を生成し送信することによって、この方法は開始される（ステップ２１０）。サーバは、このソフトウエアアップデート要求を受信し、クライアントコンピュータにテストを送信することによって応答する（ステップ２２０）。本明細書で論じるように、このテストは、アップデート要求の対象であるクライアントソフトウエアアプリケーションが、正規のものであるかどうかを検出するように設計されたものである。クライアントコンピュータに送信されるテストは、要求された特定のソフトウエアアップデート、ならびにクライアントコンピュータ上で利用可能なバージョンまたはサービスパックアップグレードに関する情報に応じて適合される。そのため、本発明の一実施形態においては、ソフトウエアアップグレード要求を受信した後で、サーバとクライアントの間で一連の問い合わせ（ｑｕｅｒｙ）がなされ、それによって、サーバは、より詳細にアップグレードの範囲を確認し、クライアントコンピュータ上で利用可能なソフトウエア製品およびバージョンを識別することができる。

【００１９】

サーバは、いずれのタイプの、いずれのバージョンのソフトウエアアプリケーションアップデートが要求されているかを判定した後で、対応するテストコードをクライアントコンピュータに送信する（ステップ２２０）。クライアントコンピュータは、このテストコードを受信し、次いで、それを実行することができる（ステップ２３０）。このテストコードが実行されると、クライアントソフトウエアアプリケーションが検査され、その全体的な完全性についてテストが行われる。このテストは、クライアントのソフトウエアの一部を実行して、このコードが改ざんされていないか、また、コードがその他の点でライセンス切れになっていないかを判定するように設計することができる。コードの改変は、海賊行為が行われていることを示すことができる。同様に、コードのライセンス付与に関わるデータ（タイマ情報など）が改変されている可能性もあり、この場合も、不正な使用が行われていることを示すことができる。

【００２０】

この実行可能なテストは、クライアントコンピュータ内で実行されるので、クライアントソフトウエアコードのあらゆる面にアクセスすることができ、それによって、この対象ソフトウエアアプリケーションがアップデートの許容範囲内にあるかどうかを見極めることができる。本発明の一態様は、クライアントコンピュータの外部のソースを介して、例えばサーバによって、このようなテストコードを提供することである。このことにより、いかなるクライアントベースのコードも、クライアントソフトウエアアプリケーションの完全性を検証するために実行されるテストのタイプを予測することができない。その結果、クライアントソフトウエアアプリケーションを改変したソフトウエア盗用者は、おそら

くはこのソフトウエアアプリケーションのライセンス制限を打ち負かそうとする際の行動をすべて隠すことができなくなる。この実行可能なテストがクライアントコンピュータ上で実行された後で、この完全性テストの結果はサーバに提供される（ステップ２４０）。次いで、サーバはこのテスト結果の処理を開始する（ステップ２５０）。

【００２１】

他の実施形態として、クライアントはアップデート要求を送信し（ステップ２１０）、（点線を通って）サーバはこの要求を受信することができる（ステップ２２１）。サーバは、クライアントのソフトウエアの完全性を検証する最も適切なテストを決定することができる。上記で述べたように、サーバに対してアップグレード要求がなされた後で、サーバとクライアントの間で一連の問い合わせ（ｑｕｅｒｙ）が行われ、それによって、サーバが、より詳細にアップグレードの範囲を確認し、クライアントコンピュータ上で利用可能なソフトウエア製品およびバージョンを識別することができる。サーバによって適切な完全性テストが決定されると、サーバ自身が、この完全性テストを実行することができる（ステップ２３１）。

【００２２】

ステップ２３１を通る経路を用いると、テスト結果の処理は広範なものになり、ステップ２４０を通る経路を用いると、テスト結果の処理は比較的直接的なものになり得る（ステップ２５０）。広範な処理においては、ＣＲＣ、チェックサム、ハッシュ値、デジタル署名、隠し実行可能プログラムその他のテスト方法の処理を行い、それらの結果と、基準値および許容差とを比較して、テストの合格・不合格を判定することができる。クライアントアプリケーションソフトウエアに対して実施されるテストにより、合格または不合格のステータスがもたらされ、このステータスおよび／またはデータをサーバに送り返すことも可能である。いずれの場合でも、ステップ２６０で、合格または不合格について処理結果を検査することがきできる。

【００２３】

この結果が、テストに不合格にならなかったことを示し、その結果、クライアントソフトウエアアプリケーションが本物とみなされる場合、要求されたクライアントソフトウエアアップデートを提供する権限がサーバに与えられる（ステップ２８０）。テストに不合格だった場合には、このソフトウエアの完全性は、まず間違いなく損なわれており、要求されたソフトウエアアプリケーションのアップデートは拒否される（ステップ２７０）。

【００２４】

一実施形態においては、クライアントソフトウエアアプリケーションの完全性をテストするためにサーバからクライアントに提供されるテストは、時間ゼロの情報とともに、暗号化されたＰＩＤ（製品識別子）またはＰｋｅｙ（プロダクトキー）、あるいはＰＩＤ／Ｐｋｅｙの組合せを、サーバに送信することを含むことができる。このサーバは例えば、米国ワシントン州Ｒｅｄｍｏｎｄ所在のＭｉｃｒｏｓｏｆｔ（登録商標）社から入手可能なＷｉｎｄｏｗｓ（登録商標）アップデートサーバとすることができる。サーバが、Ｎ回（例えば、Ｎ＜１００）以上の回数にわたって、同じＰＩＤ／Ｐｋｅｙの組合せに対して異なる時間ゼロを受け取る場合、２つ以上のマシンが同じＰＩＤ／Ｐｋｅｙを使用していると結論づけることができる。このような状況下では、このクライアントソフトウエアアプリケーションは、ライセンス外で動作しているとみなすことができ、上記で説明したように、ソフトウエアのアップデートを拒否することができる。

【００２５】

別の実施形態においては、ソフトウエアに基づいてライセンスを実施することを、クライアントソフトウエアアプリケーションに組み込むことができる。対応するＡＰＩ（アプリケーションプログラミングインターフェース）を、クライアントソフトウエアアプリケーションとともに出荷される様々なバイナリファイルに実装することができる。クライアントに送信されるテストプログラムをクライアントコンピュータ上で実行し、このテストプログラムが、これらのＡＰＩを呼び出して、特定のクライアントソフトウエアアプリケーションについての情報を取得し、この情報をサーバに送信することができる。この同じ

テストプログラムが、バイナリ選択されたバイナリファイルのハッシュ値を計算し、これらをアップデートサーバに送信することもできる。次いで、これらのハッシュ値と、サーバ側のテーブルにある有効なバイナリハッシュ値とを比較する。これらのハッシュ値が一致しない場合、高い確率で、このバイナリファイルにパッチが当てられているか、あるいは、このバイナリファイルは、ソフトウエア製造業者が意図しないなんらかの方法で改変されている。この実施形態においては、ソフトウエア製造業者によって正規のソフトウエアアップグレードまたはサービスパックアップデートが公開されるたびにハッシュテーブル内のエントリを更新する仕組みを、アップデートサーバの内部または外部に設けることができる。

【００２６】

別の実施形態においては、ＸｒＭＬライセンスファイルなどのアクティブなライセンスファイルのハッシュ値または実際の値をアップデートサーバに送信することができる。これらのアクティブなライセンスファイルまたはＸｒＭＬライセンスは、変更が生じる可能性が低いので、アップデートサーバに送信するバイナリファイルの更新されるハッシュ値の更新回数を少なくすることができる。

【００２７】

別の実施形態においては、アップデートを要求された場合、クライアントソフトウエアアプリケーション自体の中でハートビートシーケンス（ｈｅａｒｔ－ｂｅａｔ　ｓｅｑｕｅｎｃｅ）を生成できる自己評価機構を、アップデートサーバによって探査することができる。アップデートサーバは、このハートビートシーケンスの検出またはそのテストを行うことができ、このシーケンスの正当性を確認することができる。この正当性の確認に基づいて、上記で説明したように、要求されたアップデートを提供または拒否することができる。

【００２８】

別の実施形態においては、実際のバイパスライセンスキーを安全にアップデートサーバに送信し、次いで、完全性テストにより、この本物のキーと、クライアントコンピュータ上に存在するバイパスライセンスキーとを比較することができる。このようにして、アップデートサーバは、ライセンスキーをテストし、偽のバイパスライセンスキーを使用しているクライアントコンピュータを識別することができる。さらに、偽物のプロダクトキーを検出し、本物のプロダクトキーと区別して、海賊版クライアントソフトウエアを識別することができる。

【００２９】

ネットワークを使用しない本発明の実施形態においては、上記で説明したサーバ側認証機構を、本発明におけるクライアントにより実行されるテストとして使用することができ、また、署名されたバイナリファイルに統合することもできる。このバイナリファイルは、ＣＤの形で出荷されるオフラインソフトウエアパッケージのインストーラと統合することができる。この実施形態においては、顧客は、ソフトウエアに対するアップデートを含むＣＤを購入することになる。ＣＤ上のこのインストーラは、アップグレードおよび／または新しいアップデートの適用前に、認証機構を呼び出して、クライアントマシン上にすでにインストールされているクライアントソフトウエアを検証することができる。テストの仕組みを備えたＣＤが、クライアントコンピュータに、完全性が損なわれたアプリケーションが含まれることを検出した場合、このアプリケーションのアップデートは拒否されることになる。このテストの仕組みが、クライアントソフトウエアアプリケーションが本物であることを検出した場合、このアプリケーションに対するアップデートが提供されることになる。

【００３０】

コンピューティング装置の例

図３および以下の考察は、本発明の実施形態を実施しできる適切なコンピューティング環境を、簡単かつ全体的に説明するためのものである。以下、汎用コンピュータを説明するが、これは一例であり、ネットワーク／バスによる相互運用および相互作用が可能なク

ライアントなど、他のコンピューティング装置で本発明の実施形態を実施することができる。そのため、本発明の実施形態は、関与するクライアントリソースが極めて少ない、または最小限であるネットワーク化されたホストサービス環境で実施できる。このような環境は、ネットワーク／バスへのインターフェースとして働くネットワーク化された環境である。例えば、クライアント装置が単に、機器内に設けられたオブジェクト、または他のコンピューティング用の装置およびオブジェクトなどがある。本質的には、データを記憶させることができ、また、データを取り出すことができる場所ならどこでも、望ましいまたは適切な動作環境になる。

【００３１】

要件ではないが、本発明の実施形態は、装置またはオブジェクト用のサービスの開発者が使用し、かつ／または、アプリケーションソフトウエアに含まれるオペレーティングシステムによって実施することもできる。ソフトウエアは、クライアントワークステーション、サーバその他の装置など、１つまたは複数のコンピュータによって実行されるコンピュータが実行可能な命令、例えばプログラムモジュールの一般の状況において説明できる。一般に、プログラムモジュールは、特定のタスクを実施し、また特定の抽象データタイプを実装するルーチン、プログラム、オブジェクト、コンポーネント、データ構造などを含む。典型的には、これらのプログラムモジュールの機能は、様々な実施形態において、所望のとおりに組み合わせるか、あるいは分散させることができる。さらに、当業者には、他のコンピュータ構成によって本発明の様々な実施形態を実施できることが理解されよう。使用に適した他の周知のコンピューティングシステム、環境、および／または構成には、ＰＣ（パーソナルコンピュータ）、現金自動預入払出機、サーバコンピュータ、ハンドヘルドまたはラップトップ装置、マルチプロセッサシステム、マイクロプロセッサベースのシステム、プログラム可能な民生用電子機器、ネットワークＰＣ、機器、光源、環境制御要素、ミニコンピュータ、大型コンピュータなどが含まれるが、これらに限定されるものではない。本発明の実施形態は、通信ネットワーク／バスその他のデータ送信メディアを介して接続された遠隔処理装置によってタスクが実施される分散コンピューティング環境で実施することもできる。分散コンピューティング環境では、プログラムモジュールは、メモリ記憶装置を含めて、ローカルおよびリモートコンピュータ記憶メディアに配置することができ、クライアントノードは、サーバノードとして挙動することができる。

【００３２】

図３は、本発明の実施形態を実施できる適切なコンピューティングシステム環境の実施例７００を示す。ただし、上記で明確にしたように、コンピューティングシステム環境７００は、適切なコンピューティング環境の単なる１つの例であり、本発明の実施形態の利用法または機能の範囲に関していかなる制限も示唆するものではない。また、コンピューティング環境７００を、動作環境の実施例７００に示すコンポーネントのいずれか１つまたはそれらの組合せに関係する依存性または要件を持っていると解釈すべきではない。

【００３３】

図３を参照すると、本発明の実施形態を実施するシステムは、コンピュータシステム７１０の形態で汎用コンピューティング装置を含む。コンピュータシステム７１０のコンポーネントは、処理装置７２０と、システムメモリ７３０と、システムメモリを含めて様々なシステムコンポーネントを処理装置７２０に接続するシステムバス７２１を含むことができるが、これらに限定されるものではない。システムバス７２１は、メモリバスまたはメモリコントローラ、ペリフェラルバス、およびローカルバスを含めて、様々なバスアーキテクチャのいずれかを利用するいくつかのタイプのバス構造のいずれかとすることができる。例を挙げると、このようなアーキテクチャには、ＩＳＡ（業界標準アーキテクチャ）バス、ＭＣＡ（マイクロチャネルアーキテクチャ（Ｍｉｃｒｏ Ｃｈａｎｎｅｌ Ａｒｃｈｉｔｅｃｔｕｒｅ））バス、ＥＩＳＡ（拡張ＩＳＡ）バス、ＶＥＳＡ（ビデオ電子規格協会）ローカルバス、および（メザニン（Ｍｅｚｚａｎｉｎｅ）バスとしても知られる）ＰＣＩ（ペリフェラルコンポーネントインターコネクト（Ｐｅｒｉｐｈｅｒａｌ Ｃｏｍｐｏｎｅｎｔ Ｉｎｔｅｒｃｏｎｎｅｃｔ））バスが含まれるが、これらに限定される

ものではない。

【００３４】

コンピュータシステム７１０は一般に、様々なコンピュータ読取り可能媒体を含む。コンピュータ読取り可能媒体は、コンピュータシステム７１０がアクセスできる任意の利用可能な媒体とすることができ、揮発性および不揮発性メディア、リムーバブルおよび非リムーバブルメディアをともに含む。例を挙げると、コンピュータ読取り可能媒体は、コンピュータ記憶媒体および通信媒体を含み得るが、これらに限定されるものではない。コンピュータ記憶媒体は、コンピュータ可読命令、データ構造、プログラムモジュールその他のデータなどの情報を記憶するための任意の方法または技術で実施される揮発性および不揮発性メディア、リムーバブルおよび非リムーバブルメディアを含む。コンピュータ記憶媒体は、ＲＡＭ（ランダムアクセスメモリ）、ＲＯＭ（読出し専用メモリ）、ＥＥＰＲＯＭ（電気的に消去可能なプログラマブル読出し専用メモリ）、フラッシュメモリその他のメモリ技術、ＣＤＲＯＭ（コンパクトディスク型読出し専用メモリ）、ＣＤＲＷ（再書込み可能なコンパクトディスク）、ＤＶＤ（デジタル多用途ディスク）その他の光ディスク記憶装置、磁気カセット、磁気テープ、磁気ディスク記憶装置その他の磁気記憶装置、あるいは所望の情報を記憶するのに使用することができ、かつコンピュータシステム７１０がアクセスできる他のメディアを含むが、これらに限定されるものではない。通信媒体は一般に、搬送波その他の搬送機構などの変調データ信号の形で、コンピュータ可読命令、データ構造、プログラムモジュールその他のデータを実施するものであり、任意の情報送達メディアを含む。「変調データ信号」という用語は、その信号に情報を符号化するようにその信号の特性の１つまたは複数が設定または変更された信号を意味する。例を挙げると、通信メディアは、有線ネットワークまたは直接有線接続などの有線メディア、および音波、ＲＦ、赤外線その他の無線メディアなどの無線メディアを含むが、これらに限定されるものではない。上記のいずれの組合せも、コンピュータ可読メディアの範囲に含めるべきである。

【００３５】

システムメモリ７３０は、ＲＯＭ（読出し専用メモリ）７３１およびＲＡＭ（ランダムアクセスメモリ）７３２など、揮発性および／または不揮発性メモリの形態でコンピュータ記憶媒体を含む。例えば起動時に、コンピュータシステム７１０内の要素間で情報を転送する助けとなる基本ルーチンを含むＢＩＯＳ（基本入出力システム）７３３は一般に、ＲＯＭ７３１内に格納される。ＲＡＭ７３２は一般に、処理装置７２０が、即座にアクセスすることができ、かつ／または現在操作中のデータおよび／またはプログラムモジュールを含む。例を挙げると、図３に、オペレーティングシステム７３４、アプリケーションプログラム７３５、他のプログラムモジュール７３６、およびプログラムデータ７３７を示すが、これらに限定されるものではない。

【００３６】

コンピュータシステム７１０は、他のリムーバブル／非リムーバブルな、揮発性／不揮発性のコンピュータ記憶媒体も含み得る。単なる例として、図３に、不揮発性非リムーバブル磁気メディアに対して読書きを行うハードディスクドライブ７４１、不揮発性リムーバブル磁気ディスク７５２に対して読書きを行う磁気ディスクドライブ７５１、およびＣＤ　ＲＯＭ、ＣＤＲＷ、ＤＶＤその他の光メディアなどの不揮発性リムーバブル光ディスク７５６に対して読書きを行う光ディスクドライブ７５５を示す。例示の動作環境で使用できる他のリムーバブル／非リムーバブルな、揮発性／不揮発性のコンピュータ記憶媒体には、磁気テープカセット、フラッシュメモリカード、デジタル多用途ディスク、デジタルビデオテープ、ソリッドステートＲＡＭ、ソリッドステートＲＯＭなどが含まれるが、これらに限定されるものではない。ハードディスクドライブ７４１は一般に、インターフェース７４０などの非リムーバブルメモリ用インターフェースを介してシステムバス７２１に接続され、磁気ディスクドライブ７５１および光ディスクドライブ７５５は一般に、インターフェース７５０などのリムーバブルメモリ用インターフェースによってシステムバス７２１に接続される。

【００３７】
上記で論じ、かつ図３に示すドライブおよびそれらに関連するコンピュータ記憶媒体は、コンピュータシステム７１０用のコンピュータ可読命令、データ構造、プログラムモジュールその他のデータの記憶装置を提供する。図３では、例えば、ハードディスクドライブ７４１は、オペレーティングシステム７４４、アプリケーションプログラム７４５、他のプログラムモジュール７４６、およびプログラムデータ７４７を記憶するように示されている。これらのコンポーネントは、オペレーティングシステム７３４、アプリケーションプログラム７３５、他のプログラムモジュール７３６、およびプログラムデータ７３７と同じものとすることもできるし、異なるものとすることもできることに留意されたい。ここでは、オペレーティングシステム７４４、アプリケーションプログラム７４５、他のプログラムモジュール７４６、およびプログラムデータ７４７には、少なくともそれらが異なるコピーであることを示すために、異なる番号が与えられている。ユーザは、キーボード７６２、および一般にマウス、トラックボールまたはタッチパッドと呼ばれるポインティングデバイス７６１などの入力装置を介して、コンピュータシステム７１０にコマンドおよび情報を入力することができる。（図示しない）他の入力装置は、マイクロホン、ジョイスティック、ゲームパッド、衛星放送受信アンテナ、スキャナなどを含むことができる。上記その他の入力装置は、システムバス７２１に結合されたユーザ入力インターフェース７６０を介して処理装置７２０に接続されることが多いが、パラレルポート、ゲームポート、またはＵＳＢ（ユニバーサルシリアルバス）など他のインターフェースおよびバス構造によって接続することもできる。モニタ７９１その他のタイプのディスプレイ装置も、（図示しない）ビデオメモリと通信し得るビデオインターフェース７９０などのインターフェースを介してシステムバス７２１に接続される。コンピュータシステムは、モニタ７９１に加えて、出力用周辺インターフェース７９５を介して接続できるスピーカ７９７およびプリンタ７９６など他の周辺出力装置も含むことができる。

【００３８】
コンピュータシステム７１０は、リモートコンピュータ７８０など、１つまたは複数のリモートコンピュータへの論理接続部を利用するネットワーク環境または分散環境で動作することができる。リモートコンピュータ７８０は、パーソナルコンピュータ、サーバ、ルータ、ネットワークＰＣ、ピアデバイス（ｐｅｅｒ　ｄｅｖｉｃｅ）その他一般のネットワークノードとすることができる。一般に、コンピュータシステム７１０に関連して上記で説明した要素の多くまたはすべてを含むが、図３にはメモリ記憶装置７８１だけを示す。図３に示す論理接続部は、ＬＡＮ（ローカルエリアネットワーク）７７１およびＷＡＮ（ワイドエリアネットワーク）７７３を含むが、他のネットワーク／バスも含み得る。このようなネットワーク環境は、家庭、一般事務所、企業規模のコンピュータネットワーク、イントラネット、およびインターネットで一般的なものである。

【００３９】
ＬＡＮネットワーク環境で用いられるとき、コンピュータシステム７１０は、ネットワークインターフェースまたはアダプタ７７０を介してＬＡＮ７７１に接続される。ＷＡＮネットワーク環境で用いられるとき、コンピュータシステム７１０は一般に、インターネットなどのＷＡＮ７７３を介して通信を確立するためのモデム７７２その他の手段を含む。内蔵型または外付けとできるモデム７７２は、ユーザ入力インターフェース７６０その他の適切な機構を介してシステムバス７２１に接続することができる。ネットワーク環境においては、コンピュータシステム７１０に関連して示すプログラムモジュールまたはその一部は、リモートメモリ記憶装置に格納することができる。例を挙げると、図３に、メモリ装置７８１上に常駐するようにリモートアプリケーションプログラム７８５を示すが、これに限定されるものではない。図３に示すネットワーク接続部は例であり、コンピュータ間で通信リンクを確立する他の手段を使用できることを理解されたい。

【００４０】
様々な分散コンピューティング構成が、パーソナルコンピューティングとインターネットを統合させるという観点からこれまで開発されており、現在も開発中である。個人およ

び企業のユーザには等しく、アプリケーションおよびコンピューティング装置用のシームレスに相互運用可能なウェブ対応型インターフェースが提供され、それによって、コンピューティング活動が、ますますウェブブラウザ指向またはネットワーク指向になる。

【００４１】

例えば、Ｍｉｃｒｏｓｏｆｔ社から入手可能なＭＩＣＲＯＳＯＦＴ（登録商標）．ＮＥＴ（商標）プラットホームは、サーバ、ウェブベースのデータ記憶などのビルディングブロックサービス（ｂｕｉｌｄｉｎｇ－ｂｌｏｃｋ　ｓｅｒｖｉｃｅｓ）、およびダウンロード可能なデバイスソフトウエアを含む。本明細書においては実施形態の例を、コンピューティング装置上に常駐するソフトウエアに関連して説明したが、オペレーティングシステム、ＡＰＩ（アプリケーションプログラミングインターフェース）、あるいは、コプロセッサ、ディスプレイ装置、および要求オブジェクトのいずれかの間の「仲介」オブジェクトによって本発明の実施形態の一部または複数の部分を実施することもでき、それによって、すべての．ＮＥＴ（商標）の言語およびサービスによって動作が実施され、それらの言語およびサービス内で動作がサポートされ、また、それらの言語およびサービスを介して動作にアクセスすることができ、また、他の分散コンピューティング構成でも同様である。

【００４２】

上記で述べたように、様々なコンピューティング装置およびネットワークアーキテクチャに関連して本発明の実施形態の例を説明してきた。その基礎となる概念は、不正なソフトウエアを検出する技術を実施することが望ましい任意のコンピューティング装置またはシステムに適用できる。そのため、本発明の実施形態に関連して説明した方法およびシステムは、様々な応用例および装置に適用できる。本明細書においては、例示のプログラミング用の言語、名称、および例を、様々な選択肢の代表として選択したが、これらの言語、名称、および例は、限定するためのものではない。本発明の実施形態によって実現されるのと同じ、または類似の、あるいは等価なシステムおよび方法を実現するオブジェクトコードを提供する多数のやり方があることが当業者には理解されよう。

【００４３】

本明細書で説明した様々な技術は、ハードウエアまたはソフトウエアに関連づけて、あるいは、適切な場合には、両者の組合せとともに実施できる。そのため、本発明の方法および機器、あるいは本発明のある種の態様またはその一部は、フロッピー（登録商標）ディスク、ＣＤ－ＲＯＭ、ハードドライブその他の任意の機械可読記憶メディアなど、有形のメディアで実施されるプログラムコード（すなわち命令）の形態をとり得る。このプログラムコードが、コンピュータなどの機械にロードされ、その機械によって実行されると、この機械は、本発明を実施する機器になる。プログラム可能なコンピュータ上でプログラムコードを実行する場合、このコンピューティング装置は一般に、プロセッサ、（揮発性および不揮発性のメモリおよび／または記憶素子を含めて）このプロセッサによって読込み可能な記憶メディア、少なくとも１つの入力装置、および少なくとも１つの出力装置を含む。本発明の実施形態の信号処理サービスを、例えば、データ処理ＡＰＩなどを使用することによって利用できる１つまたは複数のプログラムは、好ましくは、コンピュータと通信するために、高級処理型言語またはオブジェクト指向のプログラミング言語で実施される。ただし、これら１つ（または複数）のプログラムは、所望の場合にはアセンブリまたは機械語で実施できる。いずれの場合でも、この言語を、コンパイル型またはインタプリタ型の言語とし、ハードウエアの実施形態と組み合わせることができる。

【００４４】

様々な形態の好ましい実施形態に関して本発明の態様を説明してきたが、本発明から逸脱することなく本発明の同じ機能を実施するために、他の類似の実施形態を用いることもできる。上記で説明した実施形態に改変および追加を施すこともできることを理解されたい。さらに、ハンドヘルド型装置用のオペレーティングシステムおよび他の特定用途向けオペレーティングシステムを含めて、特に、無線ネットワーク装置の数が急増するにつれ、様々なコンピュータプラットホームが企図されていることを強調しておく。したがって

、特許請求する本発明は、いかなる単一の実施形態にも限定されるべきではなく、添付の特許請求の範囲による広さおよび範囲の中で解釈すべきである。

【図面の簡単な説明】

【００４５】

【図１】本発明の実施形態を実施できるネットワークのブロック図である。

【図２】本発明の態様を含む方法のフロー図である。

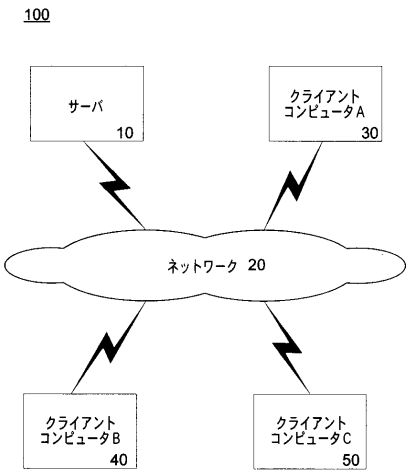
【図３】本発明の態様を実施できるコンピューティング環境の例を示すブロック図である。

【符号の説明】

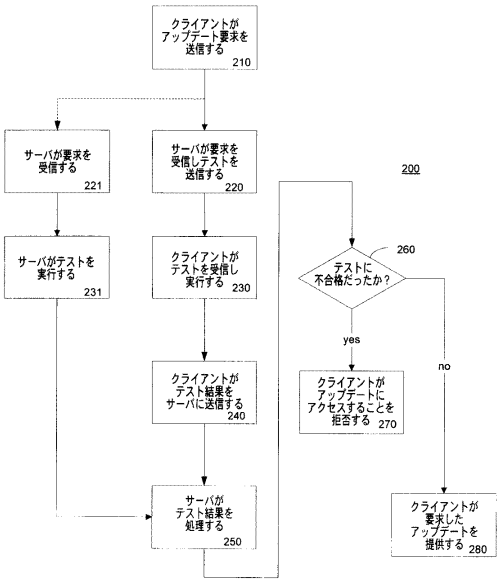
【００４６】

１００　ネットワーク構成
７１０　コンピュータシステム
７５２　不揮発性リムーバブル磁気ディスク
７５５　光ディスクドライブ
７５６　不揮発性リムーバブル光ディスク

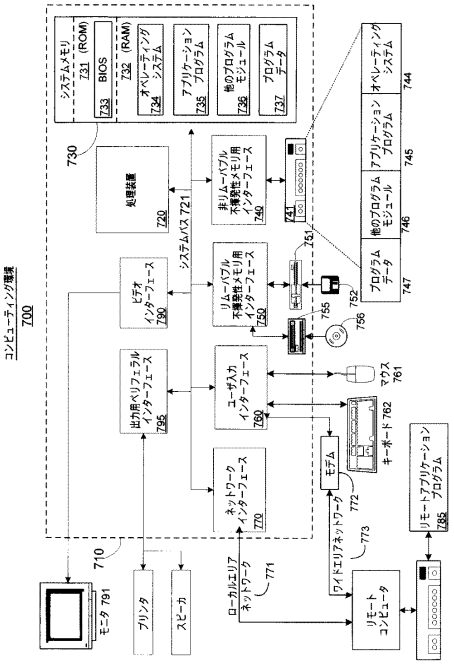
【図１】



【図２】



【図３】


コンピューティング環境
700
システムメモリ
731
(ROM)
BIOS
733
(RAM)
732
オペレーティング システム
734
アプリケーション プログラム
735
他のプログラム モジュール
736
プログラム データ
737
処理装置
720
システムバス 721
ビデオ インターフェース
790
出力周辺インターフェース
795
ユーザ入力 インターフェース
760
ネットワーク インターフェース
770
リムーバブル 不揮発性メモリ用 インターフェース
750
非リムーバブル 不揮発性メモリ用 インターフェース
740
730
710
モニタ 791
プリンタ
スピーカ
ローカルエリア ネットワーク
771
ワイドエリアネットワーク
773
モデム
772
リモート コンピュータ
マウス 761
キーボード 762
リモートアプリケーション プログラム
785
オペレーティング システム
744
アプリケーション プログラム
745
他のプログラム モジュール
746
プログラム データ
747
741
751
752
755
756

フロントページの続き

(72)発明者　クリスチャン　エリック　ハトルリッド
アメリカ合衆国　９８０５２　ワシントン州　レッドモンド　ワン　マイクロソフト　ウェイ　マイクロソフト　コーポレーション内

審査官　後藤　彰

(56)参考文献　特開２００５－３０９７５９（ＪＰ，Ａ）
特開平０９－１１４６５６（ＪＰ，Ａ）
特開平０６－３０９２６１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　２１／２２