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

特許第6399763号
(P6399763)

(45)発行日　平成30年10月3日(2018.10.3)　(24)登録日　平成30年9月14日(2018.9.14)

| (51)Int.Cl. | | F I | | |
|---|---|---|---|---|
| G06F 21/64 | (2013.01) | G06F | 21/64 | |
| H04L 9/32 | (2006.01) | H04L | 9/00 | 675B |
| H04N 1/00 | (2006.01) | H04N | 1/00 | C |
| B41J 29/38 | (2006.01) | B41J | 29/38 | Z |
| B41J 29/00 | (2006.01) | B41J | 29/00 | Z |

請求項の数 9　(全 20 頁)

(21)出願番号　特願2014-29967(P2014-29967)
(22)出願日　平成26年2月19日(2014.2.19)
(65)公開番号　特開2015-156055(P2015-156055A)
(43)公開日　平成27年8月27日(2015.8.27)
審査請求日　平成29年2月10日(2017.2.10)

(73)特許権者　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(74)代理人　100130409
弁理士　下山　治
(74)代理人　100134175
弁理士　永川　行光

最終頁に続く

(54)【発明の名称】情報処理装置、情報処理方法

(57)【特許請求の範囲】
【請求項１】
情報処理装置であって、
前記情報処理装置のシステムファイルを保持するメモリ領域に対する改竄検知を行うのか、前記システムファイルに対する改竄検知を行うのか、を設定する設定手段と、
前記設定手段による設定に応じて、前記メモリ領域に対する改竄検知若しくは前記システムファイルに対する改竄検知を行う改竄検知手段と
を備え、
前記改竄検知手段が前記システムファイルに対する改竄検知に要した時間が、前記メモリ領域に対する改竄検知に要する時間として予め定められた時間を超えた場合には、前記設定手段は、前記メモリ領域に対する改竄検知を行う、と設定する
ことを特徴とする情報処理装置。
【請求項２】
前記改竄検知手段は、前記設定手段による設定に応じて、前記メモリ領域に対応するハッシュ値を用いて前記メモリ領域に対する改竄検知を行う、若しくは前記システムファイルに対応するハッシュ値を用いて前記システムファイルに対する改竄検知を行う、ことを特徴とする請求項１に記載の情報処理装置。
【請求項３】
前記改竄検知手段は、前記メモリ領域内のシステムファイルが更新されるたびに、該更新の後の該メモリ領域に対応するハッシュ値を算出することを特徴とする請求項２に記載

の情報処理装置。
【請求項４】
前記改竄検知手段は、前記システムファイルの更新に用いるファイルの署名の検証に成功した場合には、システムファイルの取得の後に前記情報処理装置を再起動し、前記メモリ領域に対応するハッシュ値を算出することを特徴とする請求項２に記載の情報処理装置。
【請求項５】
前記改竄検知手段が算出したハッシュ値を暗号化してから前記メモリ領域とは異なる別メモリ領域に登録する登録手段を更に備えることを特徴とする請求項２乃至４の何れか１項に記載の情報処理装置。
【請求項６】
前記改竄検知手段は、
前記メモリ領域に対応するハッシュ値と、前記別メモリ領域に登録されているハッシュ値の復号結果と、が一致しているか否かを判断し、一致していないと判断した場合には、前記メモリ領域に対する改竄が行われたことを報知することを特徴とする請求項５に記載の情報処理装置。
【請求項７】
前記メモリ領域は前記システムファイルを保持するパーティションであることを特徴とする請求項１乃至６の何れか１項に記載の情報処理装置。
【請求項８】
情報処理装置が行う情報処理方法であって、
前記情報処理装置の設定手段が、前記情報処理装置のシステムファイルを保持するメモリ領域に対する改竄検知を行うのか、前記システムファイルに対する改竄検知を行うのか、を設定する設定工程と、
前記情報処理装置の改竄検知手段が、前記設定工程による設定に応じて、前記メモリ領域に対する改竄検知若しくは前記システムファイルに対する改竄検知を行う改竄検知工程と
を備え、
前記改竄検知工程で前記システムファイルに対する改竄検知に要した時間が、前記メモリ領域に対する改竄検知に要する時間として予め定められた時間を超えた場合には、前記設定工程では、前記メモリ領域に対する改竄検知を行う、と設定する
ことを特徴とする情報処理方法。
【請求項９】
コンピュータを、請求項１乃至７の何れか１項に記載の情報処理装置の各手段として機能させるためのコンピュータプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、改ざんの有無の検出技術に関するものである。
【背景技術】
【０００２】
近年の複合機（ＭＦＰ）はネットワークインターフェイスを有し、ファイルサーバ機能やメール送受信機能などＰＣやサーバなどとそん色のない機能を有している。このような状況において、複合機においてもＰＣやサーバと同様に不正なハッキングによる機器の不正使用が問題視されている。
【０００３】
複合機のような組み込み型の情報処理装置の場合、多くの場合は、出荷時に想定されたファームウェアだけの使用を想定している。使用可能なファームウェアの個々のファイルの検証データを予め保管し、ファームウェアの使用時に検証データを用いて該ファームウェアに対する改ざん検証を行い、正しいファームウェアだけを使用可能とするホワイトリ

スト方式が一般的である。ここでいう検証データとは、ファームウェアの一意性を確認するために、ファームウェアが格納された個々のファイルのバイナリデータから、ハッシュ関数を用いて作成されたハッシュ値を用いるのが一般的である。ハッシュ関数は一方向性関数とも呼ばれる。ハッシュ関数を用いれば、ある値からハッシュ値を作成した際に、ハッシュ値を変更せずに元の値を改変することが不可能であるという特徴があり、正しいファームウェアであるかどうかの改ざん検証に使用することが可能である。また、改ざん確認とは、ファームウェアを実行する際やファイルをオープンする際にファイル全体のハッシュ値を算出し、ホワイトリストに保管された該ファイルに対する検証データと一致しているかを確認し、一致しなければ改ざんされたと見なして警告など使用者に通告を行うことを示している。

【０００４】

通常、ファームウェアは複数のファイルから構成されるため、検証データはファイルに対応した数だけ複数保管されている。このようなホワイトリスト方式の改ざん検証を行うソフトウェアは広く一般化しており、例えばTripwireなどの製品が存在する。

【０００５】

また、ファームウェアの改ざんを防ぐために、不揮発のディスクではなく、ＲＡＭにディスクのデータをコピーして、そのＲＡＭをディスクとしてマウントする方法が提案されている（特許文献１）。

【先行技術文献】

【特許文献】

【０００６】

【特許文献１】特開2004-206394号公報

【発明の概要】

【発明が解決しようとする課題】

【０００７】

以下に従来のホワイトリスト方式の問題を挙げる。

【０００８】

第１に、膨大なソフトウェアに対する個々の検証データを機器の内部に保管する必要があり、その場合、不揮発容量が限られた機器の場合には不揮発メモリ容量を増やす必要が発生し、機器のコストを増加させる、という問題がある。

【０００９】

第２に、膨大なソフトウェアの改ざん検証を行うと、個々のファイル読み込みやハッシュ算出のオーバーヘッドが大きくなることによって機器の動作負荷が高くなってしまい、動作性能を下げてしまう、という問題がある。

【００１０】

本発明はこのような問題に鑑みてなされたものであり、メモリ領域を多分に使用することなく、且つ処理負荷を増大させることなく、多量のシステムファイルに対する改ざん検証を実現させる技術を提供する。

【課題を解決するための手段】

【００１１】

本発明の一様態は、情報処理装置であって、

前記情報処理装置のシステムファイルを保持するメモリ領域に対する改竄検知を行うのか、前記システムファイルに対する改竄検知を行うのか、を設定する設定手段と、

前記設定手段による設定に応じて、前記メモリ領域に対する改竄検知若しくは前記システムファイルに対する改竄検知を行う改竄検知手段と

を備え、

前記改竄検知手段が前記システムファイルに対する改竄検知に要した時間が、前記メモリ領域に対する改竄検知に要する時間として予め定められた時間を超えた場合には、前記設定手段は、前記メモリ領域に対する改竄検知を行う、と設定する

ことを特徴とする。

【発明の効果】
【００１２】
本発明の構成により、メモリ領域を多分に使用することなく、且つ処理負荷を増大させることなく、多量のシステムファイルに対する改ざん検証を実現させることができる。
【図面の簡単な説明】
【００１３】
【図１】システムの構成例を示すブロック図。
【図２】記憶装置１０７内の記憶領域（メモリ領域）の構成例を示す図。
【図３Ａ】システムファイルパーティション２０１内のシステムファイルの配置例を示す図。
【図３Ｂ】ハッシュ値の算出処理について説明する図。
【図４Ａ】システムファイルパーティション２０１に対する改ざん例を示す図。
【図４Ｂ】システムファイルパーティション２０１に対する改ざん例を示す図。
【図４Ｃ】システムファイルパーティション２０１に対する改ざん例を示す図。
【図５】複合機１０１の機能構成例を示すブロック図。
【図６】複合機１０１が行う処理のフローチャート。
【図７】複合機１０１が行う処理のフローチャート。
【図８】従来方式と本方式の性能の傾向を説明するグラフ。
【図９】複合機１０１が行う処理のフローチャート。
【図１０】複合機１０１が行う処理のフローチャート。
【図１１】複合機１０１が行う処理のフローチャート。
【発明を実施するための形態】
【００１４】
以下、添付図面を参照し、本発明の好適な実施形態について説明する。なお、以下説明する実施形態は、本発明を具体的に実施した場合の一例を示すもので、特許請求の範囲に記載した構成の具体的な実施例の１つである。
【００１５】
［第１の実施形態］
以下では、本実施形態に係る情報処理装置としての複合機（ＭＦＰ）と、ＰＣ（パーソナルコンピュータ）と、を有するシステムを例にとり、該複合機による、システムファイルを保持するメモリ領域に対する改ざんの有無判断の為の処理について説明する。
【００１６】
先ず、本実施形態に係るシステムの構成例について、図１のブロック図を用いて説明する。図１に示す如く、本実施形態に係るシステムは、複合機１０１とＰＣ１２５とを有しており、複合機１０１とＰＣ１２５とはネットワーク１２６を介して互いに通信可能である。このネットワーク１２６は、ＬＡＮやインターネットなどの周知のネットワークであって、有線であっても構わないし、無線であっても構わない。
【００１７】
先ず、ＰＣ１２５について説明する。ＰＣ１２５は、例えば、文書や画像などの印刷対象を作成して複合機１０１に送信し、該複合機１０１に該印刷対象を印刷させることができる。もちろん、ＰＣ１２５が複合機１０１に対して指示する内容は印刷に限るものではなく、スキャン指示やデータ保存指示など様々な指示がある。
【００１８】
ＣＰＵ１１９は、ＲＡＭ１２０に格納されているコンピュータプログラムやデータを用いて処理を実行することで、ＰＣ１２５全体の動作制御を行うと共に、ＰＣ１２５が行うものとして後述する各処理を実行する。
【００１９】
ＲＡＭ１２０は、記憶装置１２２からロードされたコンピュータプログラムやデータ、ネットワーク通信部１１８を介して複合機１０１から受信した様々なデータ、を一時的に記憶するためのエリアを有する。また、ＲＡＭ１２０は、ＣＰＵ１１９が各種の処理を実

行する際に用いるワークエリアを有する。すなわち、ＲＡＭ１２０は、各種のエリアを適宜提供することができる。

【００２０】

記憶装置１２２は、ハードディスクドライブ装置に代表される大容量情報記憶装置である。記憶装置１２２には、ＯＳ（オペレーティングシステム）や、ＰＣ１２５が行うものとして後述する各処理をＣＰＵ１１９に実行させるためのコンピュータプログラムやデータが保存されている。記憶装置１２２に保存されているコンピュータプログラムやデータは、ＣＰＵ１１９による制御に従って適宜ＲＡＭ１２０にロードされ、ＣＰＵ１１９による処理対象となる。

【００２１】

入力部１２３は、キーボードやマウスなどにより構成されており、ＰＣ１２５の操作者が操作することで、各種の指示をＣＰＵ１１９に対して入力することができる。

【００２２】

ネットワーク通信部１１８は、ＰＣ１２５をネットワーク１２６に接続するためのもので、ＰＣ１２５は、このネットワーク通信部１１８を介して、ネットワーク１２６に接続されている複合機１０１とのデータ通信を行うことができる。

【００２３】

次に、複合機１０１について説明する。

【００２４】

ＣＰＵ１０５は、ＲＡＭ１０６に格納されているコンピュータプログラムやデータを用いて処理を実行することで、複合機１０１全体の動作制御を行うと共に、複合機１０１が行うものとして後述する各処理を実行する。

【００２５】

ＲＡＭ１０６は、記憶装置１０７からロードされたコンピュータプログラムやデータ、ネットワーク通信部１０２を介してＰＣ１２５から受信した様々なデータ、を一時的に記憶するためのエリアを有する。また、ＲＡＭ１０６は、ＣＰＵ１０５が各種の処理を実行する際に用いるワークエリアを有する。すなわち、ＲＡＭ１０６は、各種のエリアを適宜提供することができる。

【００２６】

記憶装置１０７は、ハードディスクドライブ装置に代表される大容量情報記憶装置である。記憶装置１０７には、ＯＳ（オペレーティングシステム）や、複合機１０１が行うものとして後述する各処理をＣＰＵ１０５に実行させるためのコンピュータプログラムやデータが保存されている。記憶装置１０７に保存されているコンピュータプログラムやデータは、ＣＰＵ１０５による制御に従って適宜ＲＡＭ１０６にロードされ、ＣＰＵ１０５による処理対象となる。

【００２７】

印刷エンジン１０８は、ＰＣ１２５から送信された印刷データや、スキャナエンジン１１４から出力された画像データ、等を用いて、紙などの記録媒体上に印刷を行うものである。

【００２８】

スキャナエンジン１１４は、紙などの記録媒体上に記録されている画像や文字などの情報を読み取り、読み取った結果を画像データとして出力するものである。

【００２９】

ＵＩ（ユーザインターフェース）操作部１０３は、タッチパネル画面とハードキーとを有するものであり、各種の指示をＣＰＵ１０５に対して入力するために操作者が操作するものであると共に、ＣＰＵ１０５による処理結果を画像や文字などでもって表示するものである。

【００３０】

ネットワーク通信部１０２は、複合機１０１をネットワーク１２６に接続するためのもので、複合機１０１は、このネットワーク通信部１０２を介して、ネットワーク１２６に

接続されているＰＣ１２５とのデータ通信を行うことができる。
【００３１】
なお、図１に示したＰＣ１２５及び複合機１０１の構成はあくまでも一例であり、図１に示した構成に限るものではない。また、図１では説明を簡単にするために、ＰＣ１２５及び複合機１０１のそれぞれの台数を１としているが、２以上であっても構わない。
【００３２】
次に、記憶装置１０７内の記憶領域（メモリ領域）の構成例について、図２を用いて説明する。記憶装置１０７は通常、１個の記憶デバイス（ハードディスクドライブなど）から構成されるが、記憶装置１０７内の記憶領域を論理的な単位（パーティション）で分割することが可能である。図２の例では、記憶装置１０７内の記憶領域を、３つのパーティション（システムファイルパーティション２０１、設定データパーティション２０２、画像パーティション２０３）に分割している。
【００３３】
システムファイルパーティション２０１は、複合機１０１のシステムファイル、例えば、複合機１０１の基本動作に係るコンピュータプログラムやデータのファイル等、通常の利用では変更が発生しない固定ファイルを格納するためのパーティションである。
【００３４】
設定データパーティション２０２は、複合機１０１の様々な設定データを格納したり、ファイルを一時的に格納するために使用されるパーティションである。
【００３５】
画像パーティション２０３は、複合機１０１がネットワーク通信部１０２を介してＰＣ１２５から受信した印刷データや、スキャナエンジン１１４から出力された画像データなどを一時的に格納するためのパーティションである。
【００３６】
上記で述べたように、システムファイルパーティション２０１に格納されるシステムファイルは通常利用（ソフトウェア更新を除く）では変更が入る余地がないため、常に内容は同一であると想定することができる。一方、設定データパーティション２０２や画像パーティション２０３については、ユーザによる設定変更や、日々のプリントやスキャン操作によって常に書換えが発生する。
【００３７】
なお、記憶装置１０７内のパーティション分割例はこれに限るものではなく、さらに詳細にパーティションを分割しても構わないが、上記のシステムファイルパーティション２０１に相当する記憶領域が設けられていることが前提である。
【００３８】
次に、複合機１０１の機能構成例について、図５のブロック図を用いて説明する。
【００３９】
ＦＳ（ファイルシステム）アクセス部５０１は、システムファイルパーティション２０１へのアクセスを行う。
【００４０】
ハッシュ算出部５０２は、与えられたデータのハッシュ値（該データ固有の固定サイズを有し、例えば３２バイト）を、既知のハッシュ算出アルゴリズム（例えばSHA-256）に従って算出する。ハッシュ算出アルゴリズムの特徴としては、入力データから出力データを作るのは容易であるが、同じ出力データを出力するために異なる入力データを生成することが論理的に困難であるという一方向性を持っているのが特徴である。
【００４１】
暗号処理部５０３は、与えられたデータを、既知の暗号化アルゴリズム（例えばAES）に従って暗号化することで、暗号化データを生成する。なお、この暗号処理で用いる「複合機１０１固有の暗号鍵」は、予めシステムファイルパーティション２０１等に格納されており、外部から暗号鍵が読みとれない構成となっている。なお、暗号処理部５０３は、与えられた暗号化データを復号することもできる。

【００４２】
検証データ保管部５０４は、システムファイルパーティション２０１に対する改ざんの有無を検証するために使用するデータ（システムファイルパーティション２０１に対する正しいハッシュ値）を設定データパーティション２０２に登録する。
【００４３】
ソフトウェア更新部５０５は、システムファイルパーティション２０１内に格納されているソフトウェアを更新するものであり、例えば不具合や脆弱性が発覚した時に更新する。
【００４４】
改ざん確認部５０６は、システムファイルパーティション２０１に対する改ざんの有無（例えば、システムファイルパーティション２０１内のソフトウェアの改ざんや削除、不正なソフトウェアの導入）を検証する（改竄検知）ための処理を行う。
【００４５】
図５に示した各機能部は何れもコンピュータプログラムとして実装されるもので、上記のシステムファイルパーティション２０１内にシステムファイルとして格納されている。以下では、説明上、図５に示した各機能部を処理の主体として説明する場合があるが、実際には、ＣＰＵ１０５が該機能部に相当するコンピュータプログラムを実行することでその処理がなされる。
【００４６】
次に、図２に示したシステムファイルパーティション２０１内のシステムファイル（管理領域３０１は除く）の配置例について、図３Ａを用いて説明する。管理領域３０１は、システムファイルパーティション２０１内におけるそれぞれのシステムファイルの配置を管理するために使用される領域である。ブートローダ３０２は、起動時に立ち上げるプログラムを決定する。カーネル３０３は、ＯＳの中核となるものである。デバイスドライバ３０４は、ネットワーク通信部１０２やＵＩ操作部１０３を動作させるためのデバイスドライバである。常駐実行ファイル３０５は、ファームウェア３０７とは別に動作する常駐実行ファイルである。共有ライブラリ３０６は、ファームウェア３０７から参照され、実行される、分離した共有ライブラリである。ファームウェア３０７は、複合機１０１に様々な動作を行わせるものである。言語ファイル３０８は、複合機１０１が多国語に対応するためにファームウェア３０７から分離された言語ファイルである。ＨＴＭＬコンテンツ３０９は、複合機１０１がＨＴＴＰを用いて外部から参照される時に使用されるものである。
【００４７】
また、斜線で示した領域３１０は、システムファイルパーティション２０１内の未使用領域（システムファイルが配置されていない領域）を示しており、この領域３１０には、所定の値を有するデータが登録されている。
【００４８】
なお、図３Ａでは、説明をわかりやすくするために、個々のシステムファイルは１つずつとしているが、実際には複数存在する。例えば、デバイスドライバは接続されるデバイスに応じて複数存在する。
【００４９】
また、システムファイルパーティション２０１内でシステムファイルは整然とならんでいる必要はなく、図３Ａにしめすように、実際には個々のシステムファイルはばらばらに配置される。これらシステムファイルの物理的な配置は管理領域３０１によって管理されており、どこに物理的に配置されているかをファームウェア３０７が管理する必要はない。このことは同一製品、同一バージョンのファームウェア構成の場合でも、記憶装置１０７に格納されている各システムファイルの配置が全く同じである必要はないことを示す。
【００５０】
次に、複合機１０１が、システムファイルパーティション２０１に対する改ざんの有無を検証するために用いるデータを生成するために行う処理について、同処理のフローチャ

ートを示す図６を用いて説明する。なお、図６のフローチャートに従った処理は、複合機１０１を初めて起動したときや、システムファイルパーティション２０１内のシステムファイルが更新されるたびに行われるものとする。例えば、ユーザがＵＩ操作部１０３を操作してソフトウェアの更新を指示し、システムファイルパーティション２０１内のシステムファイルの更新が完了したことをＣＰＵ１０５が検知すると、ＣＰＵ１０５は、図６のフローチャートに従った処理を実行する。

【００５１】

ステップＳ６０１では、ＣＰＵ１０５は、システムファイルパーティション２０１に対して、ＲＷ（Ｒｅａｄ／Ｗｒｉｔｅ）マウントを行う。ＲＷマウントとは、システムファイルパーティション２０１に対して読み書きができるようにする手順である。通常は読み込みしか行わないのでＲＯ（Ｒｅａｄ　Ｏｎｌｙ）マウントを行うが、システムファイルの更新時だけは書き込みを行う必要があるため、ＲＷマウントを行う。

【００５２】

ステップＳ６０２では、ＣＰＵ１０５は、ネットワーク通信部１０２を介して外部機器（例えばＰＣ１２５）から送信されたシステムファイル（例えばファームウェア）を受信し、記憶装置１０７内の設定データパーティション２０２内に一時的に格納する。

【００５３】

ステップＳ６０２では、例えば、更新ファイルを保持している更新サーバに接続し、システムソフトウェアをダウンロードし、該ダウンロードしたシステムソフトウェアを設定データパーティション２０２に一時的にコピーする。更新サーバは、例えば、インターネット上に置かれたメーカが提供するサーバである。更新サーバ以外にも、ＵＳＢメモリなどの可搬デバイスからダウンロードしてもよい。すなわち、ステップＳ６０２では、システムファイルをダウンロードして設定データパーティション２０２に格納できるのであれば、ダウンロード先やダウンロード方法は問わない。

【００５４】

次に、ステップＳ６０３では、ＣＰＵ１０５は、ステップＳ６０２で設定データパーティション２０２に格納したシステムファイルの署名を検証する。即ち、システムファイルにメーカの秘密鍵を用いて署名された署名データを付与し、複合機１０１に組み込まれた証明書を用いて検証を行い、正規なシステムファイルであるかの検証を行うものであり、転送経路に不正な改ざんがあったかを確認する仕組みである。

【００５５】

そして、この検証に成功した場合には、処理はステップＳ６０４を介してステップＳ６０５に進み、失敗した場合には、ステップＳ６０４を介して、図６のフローチャートに従った処理を完了させる。

【００５６】

ステップＳ６０５では、ＣＰＵ１０５は、システムファイルパーティション２０１内のシステムファイルのアップデートを実施する。この場合のアップデートとは、ステップＳ６０２で設定データパーティション２０２に格納したシステムファイルを、システムファイルパーティション２０１内の規定の位置に移動させることに相当する。

【００５７】

ステップＳ６０６では、ＣＰＵ１０５は、ステップＳ６０５におけるアップデートが成功したか否かを判断する。この判断の結果、成功した場合には、処理はステップＳ６０７に進み、失敗した場合には、図６のフローチャートに従った処理を完了させる。

【００５８】

ステップＳ６０７では、ＣＰＵ１０５は、システムファイルパーティション２０１をＲＯ（Ｒｅａｄ　Ｏｎｌｙ）でマウントし直す（リマウント）。このタイミングでシステムファイルパーティション２０１の書き換えは完了し、以降、次の更新が行われるまで通常はシステムファイルパーティション２０１の書き換えを行うことはない。

【００５９】

ステップＳ６０８では、ＣＰＵ１０５は、システムファイルパーティション２０１内に

格納されている全てのデータを読み出す。そしてステップＳ６０９では、ＣＰＵ１０５は、ステップＳ６０８で読み出したデータを用いて１つのハッシュ値（該データのハッシュ値）を算出する。

【００６０】

ステップＳ６０９では、例えば、ＳＨＡ－２５６ハッシュ算出アルゴリズムを用いてハッシュ値を算出する。このアルゴリズムは任意のサイズの入力データに対して、３２バイトのハッシュ値を作成する。ハッシュ算出アルゴリズムの特徴としては入力データから出力データを作るのは容易であるが、同じ出力データを出力するために異なる入力データを算出することが論理的に困難であるという一方向性を持っているのが特徴である。この特徴を利用して、正規のシステムファイルパーティション２０１から構成されたハッシュ値は、不正な改ざんを行った結果から同じ値を導き出すことが現実的に不可能であるため、ハッシュ値を検証することにより改ざんの有無を検証することが可能となる。

【００６１】

ここで、システムファイルパーティション２０１内に格納されている全てのデータを用いて１つのハッシュ値（該データのハッシュ値）を算出する処理について、図３Ｂを用いて説明する。図３Ｂでは、図３Ａで示したシステムファイルパーティション２０１の左上隅の位置からラスタスキャン順に格納データを読み出している様子を示しており、４０１はすでに読み出したデータの領域である。そしてこのようにして読み出したシステムファイルパーティション２０１内のデータを用いて、対応する１つのハッシュ値を算出する。ここで、ハッシュ値算出のためには、システムファイルのデータだけでなく、システムファイルパーティション２０１内の未使用領域（システムファイルが配置されていない領域）に格納されているデータも読み出され、読み出されたデータ全て、ハッシュ値算出のために使用される。すなわち、この算出するハッシュ値には、システムファイルのデータだけでなく、この未使用領域に格納されているデータも反映されている。然るに、システムファイルだけでなく、未使用領域についても、改ざんがあれば、ハッシュ値に変化が現れることになる。また、ハッシュ値は、システムファイルパーティション２０１内のデータ配置によって一意に作られた値であり、システムファイルパーティション２０１全体のデータが１ビットでも変わると全く異なったハッシュ値を算出することになる。そのため、任意のプログラムの改ざん、削除、不正な動作を行うファイルの追加を検知することができる。

【００６２】

ここで、システムファイルパーティション２０１内のあるシステムファイルを改ざんしたことによるシステムファイルパーティション２０１の改ざん例について、図４Ａを用いて説明する。図４Ａでは、常駐実行ファイル３０５が改ざんされ、改ざん常駐実行ファイル５９９となっている。常駐実行ファイル３０５が改ざんされて改ざん常駐実行ファイル５９９となれば、プログラムサイズの変化やプログラムの書き換えが発生し、それらはシステムファイルパーティション２０１全体の書き換えとしてハッシュ値に変化を及ぼすことになる。そのため、改ざん後のシステムファイルパーティション２０１のハッシュ値は、改ざん前のシステムファイルパーティション２０１のハッシュ値とは異なる値となり、この値の変化の検知により、図４Ａのような改ざんの有無を検知することができる。

【００６３】

次に、システムファイルパーティション２０１に不正なファイルをインストール（配置）したことによるシステムファイルパーティション２０１の改ざん例について、図４Ｂを用いて説明する。不正ファイル６０１は、正規のシステムファイルとは異なる不正なファイルであり、不正ファイル６０１をシステムファイルパーティション２０１内に配置したことで、配置前における未使用領域から不正ファイル６０１の分だけ未使用領域が減ることになる。すなわち、配置の前後で未使用領域に変化が生じることになる。本実施形態では上記の通り、システムファイルのデータだけでなく、システムファイルパーティション２０１内の未使用領域に格納されているデータも、ハッシュ値算出のために使用されるので、未使用領域に変化があれば、ハッシュ値も変化することになる。然るに、ハッシュ値

の変化の検知により、図４Ｂのような改ざんの有無を検知することができる。
【００６４】
次に、システムファイルパーティション２０１内のあるシステムファイルを削除したことによるシステムファイルパーティション２０１の改ざん例について、図４Ｃを用いて説明する。図４Ｃでは、システムファイルパーティション２０１から言語ファイル３０８が削除されている。システムファイルパーティション２０１から言語ファイル３０８が削除されると、削除前における未使用領域から言語ファイル３０８の分だけ未使用領域が増えることになる。すなわち、削除の前後で未使用領域に変化が生じることになる。本実施形態では上記の通り、システムファイルのデータだけでなく、システムファイルパーティション２０１内の未使用領域に格納されているデータも、ハッシュ値算出のために使用されるので、未使用領域に変化があれば、ハッシュ値も変化することになる。然るに、ハッシュ値の変化の検知により、図４Ｃのような改ざんの有無を検知することができる。
【００６５】
このように、本実施形態では、図４Ａ～４Ｃに例示したような改ざんを検知するために、システムファイルパーティション２０１に対するハッシュ値を算出する。ここで、従来のホワイトリスト方式においては、システムファイル単位でハッシュ値を算出しており、検証データと比較する。この場合、個々のシステムファイルはパーティション内にどのように配置されても、システムファイルの内容から生成されるハッシュ値は一意の値になるため、出荷段階でメーカ側が算出したハッシュ値を信頼することができる。
【００６６】
しかし、本実施形態のように、システムファイルパーティション全体でハッシュ値を取る場合、上記で説明している通り、個々のシステムファイルの物理的配置に制約がない。そのため、個体（複合機１０１）ごとにシステムファイルパーティションの配置が異なる場合がある。例えば、工場出荷においてバージョン１．０で出荷した製品を、出荷先（ユーザ）でバージョン１．１にバージョンアップした場合と、工場で最初からバージョン１．１で出荷した場合と、では複合機１０１の動作としては全く同じでも、システムファイルパーティションの物理的配置は全く異なる可能性がある。
【００６７】
よって、システムファイルパーティション全体のハッシュ値は出荷段階では全ての機体に共通な値をメーカが予め算出することができず、個々の複合機１０１において算出する必要がある。然るに、ハッシュ値の算出は、システムファイルの正規の更新時に行われる。
【００６８】
図６に戻って、次に、ステップＳ６１０では、ＣＰＵ１０５は、ステップＳ６０９で算出したハッシュ値を暗号化する。ステップＳ６１１では、ＣＰＵ１０５は、ステップＳ６１０で暗号化されたハッシュ値のデータ（暗号化ハッシュ値データ）を、設定データパーティション２０２（システムファイルパーティション２０１とは異なる別メモリ領域）に格納する。
【００６９】
ハッシュ値を暗号化するのは、ハッシュ値を平文で格納すると、不正な攻撃によってハッシュ値を書き換えられてしまう可能性があるからである。暗号化を行うことによって、この領域を不正に書き換えることをできないようにすることが可能である。もちろん、状況によってはこの暗号化は省いても構わない。
【００７０】
次に、システムファイルパーティション２０１に対する改ざん有無の検証処理について、同処理のフローチャートを示す図７を用いて説明する。なお、図７のフローチャートに従った処理は、複合機１０１の電源投入時に行っても構わないし、ＣＰＵ１０５が、ユーザがＵＩ操作部１０３を操作して検証処理の開始指示を入力したことを検知したことに応じて実行するようにしても構わない。また、ＣＰＵ１０５は、自身が有するタイマ機能を使用して、毎週若しくは毎月の定時に図７のフローチャートに従った処理を実行するよう

にしても構わない。すなわち、図７のフローチャートに従った処理の実行タイミングは特定のタイミングに限るものではない。

【００７１】

ステップＳ７０１では、ＣＰＵ１０５は、システムファイルパーティション２０１内に格納されている全てのデータを読み出す。そしてステップＳ７０２では、ＣＰＵ１０５は、ステップＳ７０１で読み出したデータを用いて、上記のステップＳ６０９で使用したハッシュ算出アルゴリズムと同じハッシュ算出アルゴリズム（上記の例ではＳＨＡ－２５６ハッシュ算出アルゴリズム）に従って１つのハッシュ値（該データのハッシュ値）を算出する。

【００７２】

ステップＳ７０３では、ＣＰＵ１０５は、上記のステップＳ６１１で設定データパーティション２０２に格納された暗号化ハッシュ値データを、該設定データパーティション２０２から読み出す。

【００７３】

ステップＳ７０４では、ＣＰＵ１０５は、ステップＳ７０３で読み出した暗号化ハッシュ値データを復号することで、上記のステップＳ６０９で算出したハッシュ値（正解値）を復元する。

【００７４】

ステップＳ７０５では、ＣＰＵ１０５は、ステップＳ７０２で算出したハッシュ値と、ステップＳ７０４で復元したハッシュ値と、を比較してそれぞれが一致しているか否かを判断する。この判断の結果、一致していると判断した場合には、システムファイルパーティション２０１に対する改ざんは何等行われていないと判断し、図７のフローチャートに従った処理を完了させる。一方、一致していないと判断した場合には、システムファイルパーティション２０１に対して何らかの改ざんが行われたものと判断し、処理はステップＳ７０６に進む。

【００７５】

ステップＳ７０６では、ＣＰＵ１０５は、システムファイルパーティション２０１に対して改ざんがなされていたことを報知するための処理を実行する。報知方法には様々な方法が考えられ、本実施形態では如何なる報知方法を採用しても構わない。例えば、ＣＰＵ１０５はネットワーク通信部１０２を制御して、特定のユーザ（例えばシステム管理者）の端末装置に対して改ざんがあった旨が記された電子メールを送信しても構わない。また、ＵＩ操作部１０３が有するタッチパネル画面に改ざんがあった旨のメッセージを表示しても構わない。

【００７６】

次に、改ざんの有無を検証する為の処理速度について説明する。改ざんの有無の検証処理において、対象となるシステムファイルを読み込み、ハッシュ値を算出する算出時間は、複合機１０１の動作性能を低下させる。従来のファイル単位でハッシュ値を持つホワイトリスト方式と本方式の差異について以下に述べる。

【００７７】

図８は従来方式と本方式の性能の傾向を説明するグラフである。縦軸はハッシュ算出時間(t)、横軸はシステムファイルの個数(n)を示している。８０１は従来方式の算出時間の傾向を示すグラフであり、８０２は本方式の算出時間の傾向を示すグラフを示している。なお、両方式とも、システムファイルパーティションの全容量の８０％を利用している（すなわち残りの２０％は未使用領域である）ことを想定している。ハッシュ値の算出速度は下記の算出式であらわされる

ハッシュ値の算出時間＝（ファイルを開くオーバーヘッド＋ファイル読み込み時間）＋（ハッシュ値を算出するオーバーヘッド＋ファイルのハッシュ値算出時間）

図８の例でいえば、従来方式はシステムファイルパーティションの８０％のみ算出対象となるため、ファイルの読み込み時間と算出時間は８０％に収まるが、ファイルを開くオーバーヘッド（ＨＤＤの場合には磁気ヘッドの移動時間やキャッシュのミスヒットが該当

）とハッシュ値を算出するオーバーヘッド（ハッシュ関数の初期化やメモリの確保解放が該当）が存在し、システムファイルの個数ｎの増加とともにこの値の総時間が多くなり、算出性能が低下する。一方、本方式はシステムファイルパーティション全体をひとまとめとしてハッシュ値を算出するため、システムファイルの個数に関係なく算出時間は一定である。よって、システムファイルパーティション内のシステムファイルの個数が大きい場合には従来方式よりも本方式のほうが、性能が向上することがわかる。

【００７８】

このように、本実施形態によれば、システムファイルパーティションに対する改ざんの有無を検証するために予め算出するハッシュ値は、システムファイルパーティションに対して１つあればよく、しかも高々３２バイト程度のハッシュ値を管理すればよい。このため、不揮発メモリ容量を抑えることが可能となり、機器のコスト増大を防ぐことが可能となる。

【００７９】

さらに、改ざん検証対象となるソフトウェアの個数およびサイズが十分に大きい場合、ソフトウェア単位にハッシュ値を算出するよりは、全体で１個のハッシュ値を算出することによって読み込みやハッシュ値算出のオーバーヘッドを防ぐことが可能となり、動作性能の低下を抑えることが可能となる。

【００８０】

［第２の実施形態］

動作中のファームウェアの安定を図るために、設定データパーティション２０２にシステムファイルをダウンロードすると複合機１０１を再起動させ、該再起動後に該システムファイルをシステムファイルパーティション２０１に移動させる処理（アップデート処理）を行うことが考え得る。しかし、この再起動を偽装して不正にアップデートを実施し、該アップデート後のハッシュ値を正解値として算出して設定データパーティション２０２に登録してしまうと、以降の改ざん確認において不正なシステムファイルパーティションを正しいと誤認してしまう場合がある。本実施形態は、このような点に鑑みて第１の実施形態を改良した実施形態である。以下では、第１の実施形態との差分について重点的に説明し、以下で特に触れない限りは第１の実施形態と同様であるものとする。

【００８１】

本実施形態では、システムファイルパーティション２０１に対する改ざんの有無を検証するために用いるデータを生成するために行う処理として、図６のフローチャートに従った処理の代わりに、図９，１０のフローチャートに従った処理を実行する。

【００８２】

ステップＳ９０１では、ＣＰＵ１０５は、上記のステップＳ６０１と同様にして、システムファイルパーティション２０１に対して、ＲＷ（Ｒｅａｄ／Ｗｒｉｔｅ）マウントを行う。

【００８３】

ステップＳ９０２では、ＣＰＵ１０５は、上記のステップＳ６０２と同様にして、ネットワーク通信部１０２を介して外部機器（例えばＰＣ１２５）から送信されたシステムファイル（例えばファームウェア）を受信し、記憶装置１０７内の設定データパーティション２０２内に一時的に格納する。

【００８４】

ステップＳ９０３では、ＣＰＵ１０５は、上記のステップＳ６０３と同様にして、ステップＳ９０２で設定データパーティション２０２に格納したシステムファイルの署名を検証する。そして、この検証に成功した場合には、処理はステップＳ９０４を介してステップＳ９０５に進み、失敗した場合には、ステップＳ９０４を介して図９のフローチャートに従った処理を完了させる。

【００８５】

ステップＳ９０５では、ＣＰＵ１０５は、再起動中にアップデート処理を指示するためのアップデートフラグに「１」を設定する。そして更にＣＰＵ１０５は、このアップデー

トフラグに対して秘密鍵を用いて署名を行い、設定データパーティション２０２内の不図示のフラグ領域に書き込む。なお、アップデートフラグの初期値は「０」である。

【００８６】

そしてステップＳ９０６では、ＣＰＵ１０５は、複合機１０１の再起動を開始する。ＣＰＵ１０５は、ステップＳ９０６における再起動中に、図１０のフローチャートに従った処理を実行する。

【００８７】

ステップＳ１００１では、ＣＰＵ１０５は、ステップＳ９０５で設定データパーティション２０２に書き込んだアップデートフラグを読み出す。そしてステップＳ１００２では、ＣＰＵ１０５は、ステップＳ１００１で読み出したアップデートフラグに対し、証明書を用いて署名の検証を実施する。この検証に成功した場合には、処理はステップＳ１００３を介してステップＳ１００４に進み、失敗した場合には、処理はステップＳ１００３を介してステップＳ１０１１に進む。

【００８８】

ステップＳ１００４では、ＣＰＵ１０５は、上記のステップＳ６０５と同様にして、システムファイルパーティション２０１内のシステムファイルのアップデートを実施する。

【００８９】

ステップＳ１００５では、ＣＰＵ１０５は、上記のステップＳ６０６と同様にして、ステップＳ１００４におけるアップデートが成功したか否かを判断する。この判断の結果、成功した場合には、処理はステップＳ１００６に進み、失敗した場合には、処理はステップＳ１０１１に進む。

【００９０】

ステップＳ１００６では、ＣＰＵ１０５は、上記のステップＳ６０７と同様にして、システムファイルパーティション２０１をＲＯ（Ｒｅａｄ　Ｏｎｌｙ）でマウントし直す（リマウント）。

【００９１】

ステップＳ１００７では、ＣＰＵ１０５は、上記のステップＳ６０８と同様にして、システムファイルパーティション２０１内に格納されている全てのデータを読み出す。そしてステップＳ１００８では、ＣＰＵ１０５は、上記のステップＳ６０９と同様にして、ステップＳ１００７で読み出したデータを用いて１つのハッシュ値（該データのハッシュ値）を算出する。

【００９２】

ステップＳ１００９では、ＣＰＵ１０５は、上記のステップＳ６１０と同様にして、ステップＳ１００８で算出したハッシュ値を暗号化する。ステップＳ１０１０では、ＣＰＵ１０５は、上記のステップＳ６１１と同様にして、ステップＳ１００９で暗号化されたハッシュ値のデータ（暗号化ハッシュ値データ）を、設定データパーティション２０２（システムファイルパーティション２０１とは異なる別メモリ領域）に格納する。ステップＳ１０１１では、ＣＰＵ１０５は、通常の起動処理を実施する。

【００９３】

このように、本実施形態によれば、ダウンロードしたシステムファイルの署名の検証に成功した場合にのみアップデートして正解値を算出するので、不正なファームウェアによる誤った正解値の作成を防ぐことが可能となる。

【００９４】

［第３の実施形態］

図８を用いて説明したように、システムファイルの個数が十分に少ない場合、あるいはシステムファイルパーティションの利用率が低い場合には、第１の実施形態に係る構成では、かえって速度が低下する場合がある。然るに、記憶装置１０７に十分な容量があり、従来のホワイトリスト方式が許容できる場合には、第１の実施形態に係る方式及び従来のホワイトリスト方式のうち、速度の速い方式を選択的に利用してもよい。

【００９５】

図８を用いて説明したように、システムファイルパーティション全体で改ざん確認をする場合、この確認に要する時間（確認時間）Ｔｐはシステムファイルパーティションのサイズだけに依存しており、格納されているシステムファイルの個数やサイズには一切関係ないため、固定の時間となる。

【００９６】

然るに、本実施形態では、当初は従来のホワイトリスト方式で起動し、従来のホワイトリスト方式の改ざん確認時間Ｔｆを測定する。そして確認時間Ｔｆが確認時間Ｔｐより大きくなることが確認されると、従来のホワイトリスト方式からシステムファイルパーティション全体で改ざん確認する方式に切り替える。これによって、改ざん確認による性能低下を極力抑えることが可能である。本実施形態に係る複合機１０１の起動処理について、同処理のフローチャートを示す図１１を用いて説明する。

【００９７】

ステップＳ１１０１では、ＣＰＵ１０５は、設定データパーティション２０２に格納されている改ざん確認モードを取得する。改ざん確認モードは、「ファイル単位で改ざん確認するモード」（第１のモード）、「システムファイルパーティション２０１単位で改ざん確認するモード」（第２のモード）の何れか一方のモードに設定されており、初期状態では第１のモードに設定されている。

【００９８】

ステップＳ１１０２では、ＣＰＵ１０５は、ステップＳ１１０１で取得した改ざん確認モードが第１のモードに設定されているのか、第２のモードに設定されているのかを判断する。この判断の結果、第１のモードに設定されている場合には、処理はステップＳ１１０３に進み、第２のモードに設定されている場合には、処理はステップＳ１１１１に進む。

【００９９】

ステップＳ１１１１では、ＣＰＵ１０５は、第１の実施形態で説明したシステムファイルパーティション２０１単位での改ざんの有無の検出処理を行う。そして改ざんがあると判断した場合には、処理はステップＳ１１１２を介してステップＳ１１０６に進み、改ざんがないと判断した場合には、処理はステップＳ１１１４に進む。

【０１００】

ステップＳ１１０６では、ＣＰＵ１０５は、上記のステップＳ７０６と同様にして、改ざんがあったことを報知する。一方、ステップＳ１１１４では、ＣＰＵ１０５は、上記のステップＳ１０１１と同様にして、通常の起動処理を実行する。

【０１０１】

一方、ステップＳ１１０３では、ＣＰＵ１０５は、自身のタイマ機能を用いて、現在の時刻Ｔｐｒｅを取得する。そしてステップＳ１１０４では、ＣＰＵ１０５は、従来のホワイトリスト方式に従って、システムファイル単位での改ざんの有無の検出処理を行う。

【０１０２】

そして改ざんがあると判断した場合には、処理はステップＳ１１０５を介してステップＳ１１１３に進み、改ざんがないと判断した場合には、処理はステップＳ１１０５を介してステップＳ１１０７に進む。ステップＳ１１１３では、ＣＰＵ１０５は、上記のステップＳ７０６と同様にして、改ざんがあったことを報知する。

【０１０３】

ステップＳ１１０７では、ＣＰＵ１０５は、自身のタイマ機能を用いて、現在の時刻Ｔｐｏｓｔを取得する。そしてステップＳ１１０８では、ＣＰＵ１０５は、従来のホワイトリスト方式に従って行った、システムファイル単位での改ざんの有無の検出処理に要した時間Ｔｆを、Ｔｆ＝（Ｔｐｏｓｔ－Ｔｐｒｅ）を計算することで求める。

【０１０４】

ステップＳ１１０９では、ＣＰＵ１０５は、Ｔｆ＞Ｔｐであるか否かを判断し、Ｔｆ＞Ｔｐであれば、処理はステップＳ１１１０に進み、Ｔｆ≦Ｔｐであれば、処理はステップＳ１１１４に進む。ここで、Ｔｐは常に固定であるため、複合機１０１のメーカ等が定数

として予め定義して設定データパーティション２０２に登録している。ステップＳ１１１０では、ＣＰＵ１０５は、改ざん確認モードを第２のモードに切り替える。

【０１０５】

このように、本実施形態によれば、２つの改ざん確認方式のうち、より早く改ざん確認を行うことのできる方式を利用するため、改ざん確認による速度低下を低く抑えることが可能となる。

【０１０６】

なお、図１１のフローチャートに従った処理は、複合機１０１の起動時に行われる処理として説明したが、ステップＳ１１０１～Ｓ１１１３の処理の実行タイミングは、他のタイミングであっても構わない。例えば、操作者がＵＩ操作部１０３を操作して改ざんの有無の判断指示を入力したことをＣＰＵ１０５が検知したことに応じてステップＳ１１０１～Ｓ１１１３の処理を実行しても構わない。また、予め設定された予約日時にステップＳ１１０１～Ｓ１１１３の処理を実行するようにしても構わない。

【０１０７】

［第４の実施形態］

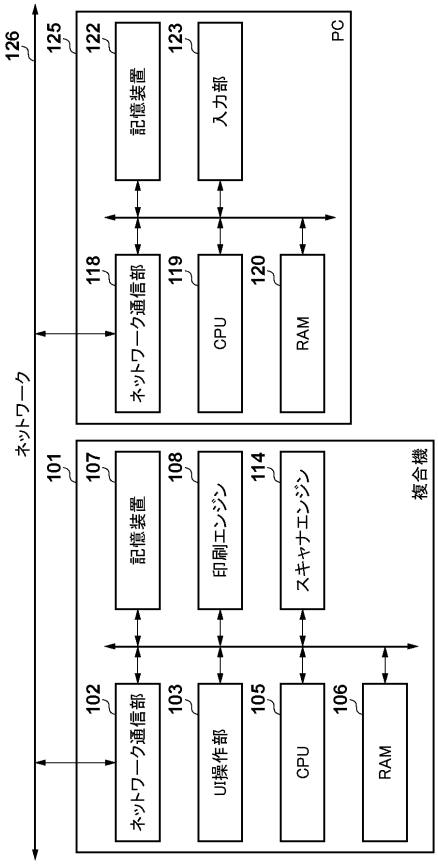
第１乃至３の実施形態では、複合機１０１における「システムファイルを保持するメモリ領域に対する改ざんの有無判断」について説明した。しかし、図６，７に示すような処理は、システムファイルパーティション２０１に相当するメモリ領域を有する装置であれば、実行可能であるし、実行することに意義はある。然るに、システムファイルを保存するためのメモリ領域内に保存されている全てのデータを用いて１つのハッシュ値を算出し、該ハッシュ値をメモリ領域とは異なる別メモリ領域に登録する、という処理（第１の処理）は複合機１０１以外の装置が実行しても構わない。また、メモリ領域内に保存されているデータを用いて１つのハッシュ値を算出し、該算出したハッシュ値と、別メモリ領域に登録されているハッシュ値と、が一致しているか否かを判断し、一致していないと判断した場合に、メモリ領域に対する改ざんが行われたことを報知する、という処理（第２の処理）についても同様で、複合機１０１以外の装置が行っても構わない。また、第１の処理と第２の処理とは同じ装置が実行するようにしても構わないし、それぞれ異なる装置が実行するようにしても構わない。

【０１０８】

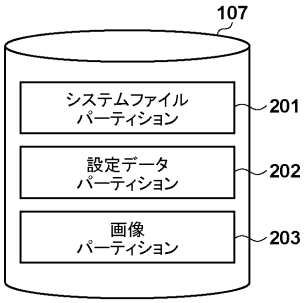
（その他の実施例）

また、本発明は、以下の処理を実行することによっても実現される。即ち、上述した実施形態の機能を実現するソフトウェア（プログラム）を、ネットワーク又は各種記憶媒体を介してシステム或いは装置に供給し、そのシステム或いは装置のコンピュータ（またはＣＰＵやＭＰＵ等）がプログラムを読み出して実行する処理である。

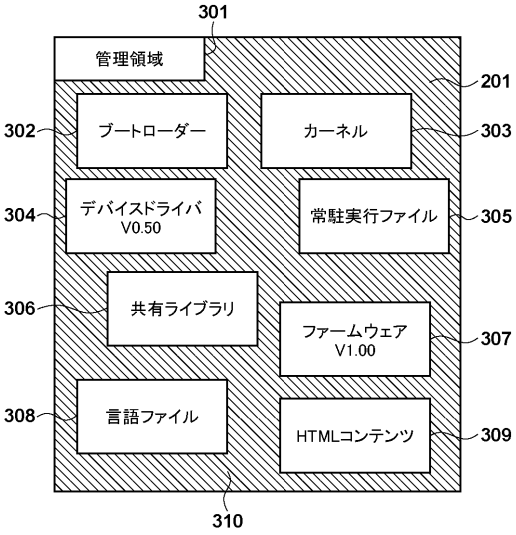
【図１】


126
125
122
記憶装置
123
入力部
PC
118
ネットワーク通信部
119
CPU
120
RAM
ネットワーク
101
107
記憶装置
108
印刷エンジン
114
スキャナエンジン
複合機
102
ネットワーク通信部
103
UI操作部
105
CPU
106
RAM

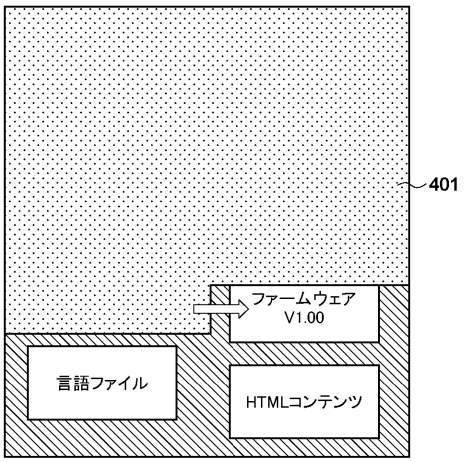
【図２】


107
システムファイル
パーティション
201
設定データ
パーティション
202
画像
パーティション
203

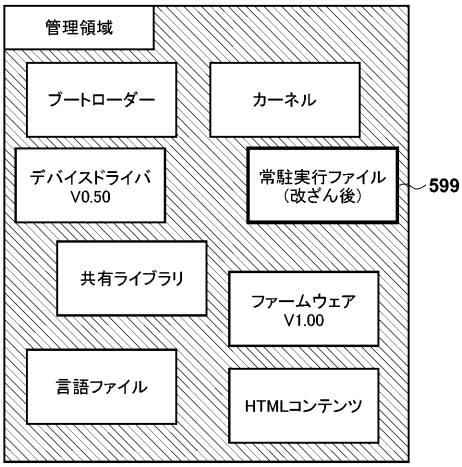
【図３Ａ】


301
管理領域
201
302
ブートローダー
カーネル
303
304
デバイスドライバ
V0.50
常駐実行ファイル
305
306
共有ライブラリ
ファームウェア
V1.00
307
308
言語ファイル
HTMLコンテンツ
309
310

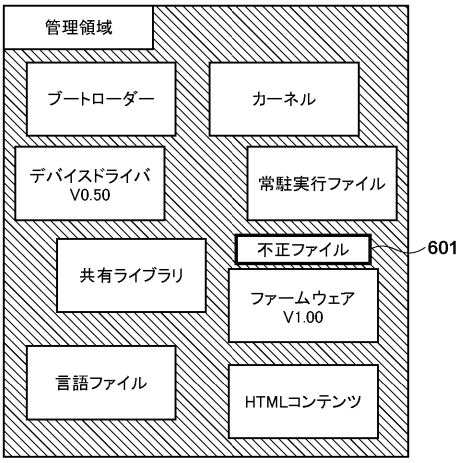
【図３Ｂ】


401
ファームウェア
V1.00
言語ファイル
HTMLコンテンツ

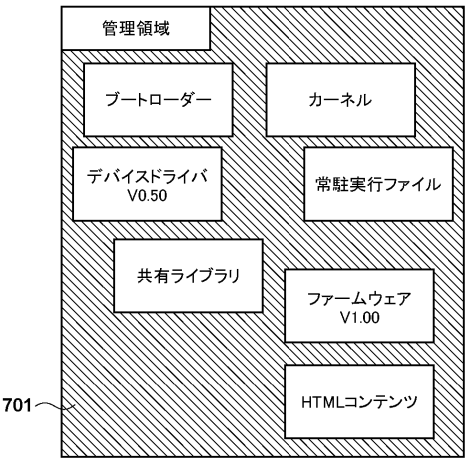
【図４Ａ】



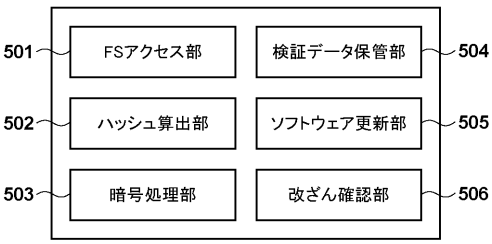

【図４Ｂ】



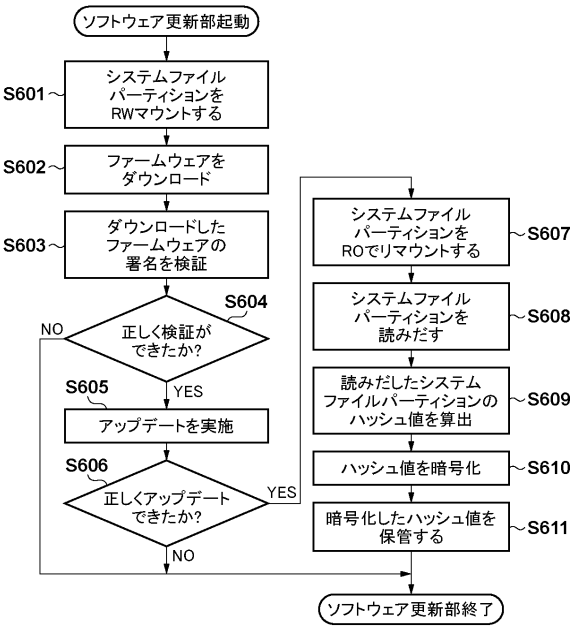

【図４Ｃ】



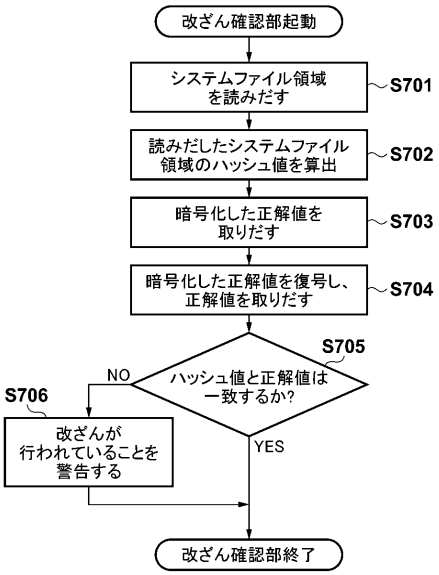

【図５】



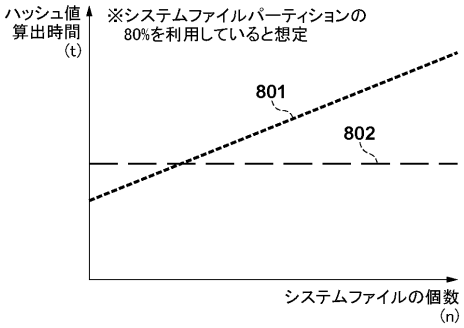

【図６】



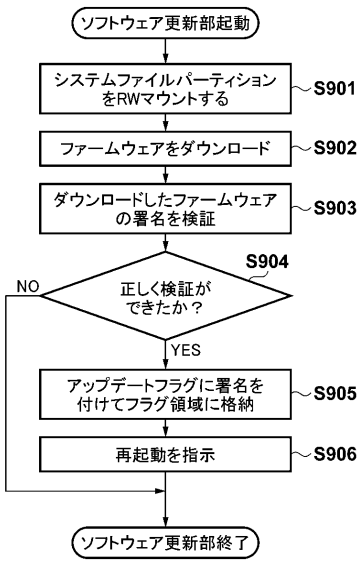
【図７】


改ざん確認部起動
システムファイル領域を読みだす
S701
読みだしたシステムファイル領域のハッシュ値を算出
S702
暗号化した正解値を取りだす
S703
暗号化した正解値を復号し、正解値を取りだす
S704
S705
ハッシュ値と正解値は一致するか?
NO
YES
S706
改ざんが行われていることを警告する
改ざん確認部終了

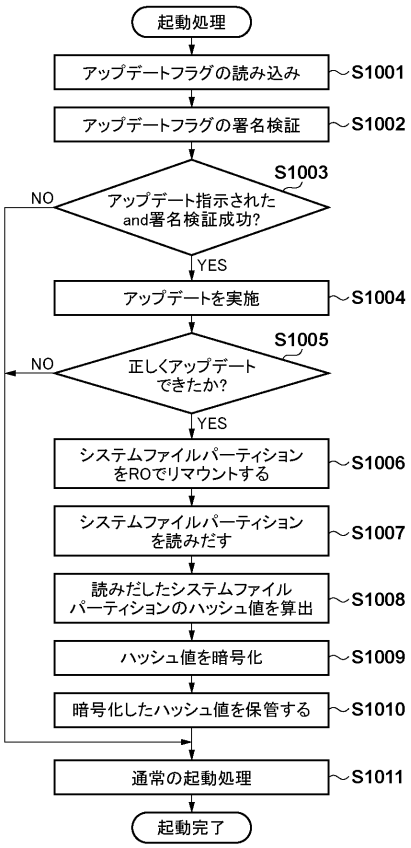
【図８】


ハッシュ値算出時間(t)
※システムファイルパーティションの80%を利用していると想定
801
802
システムファイルの個数(n)

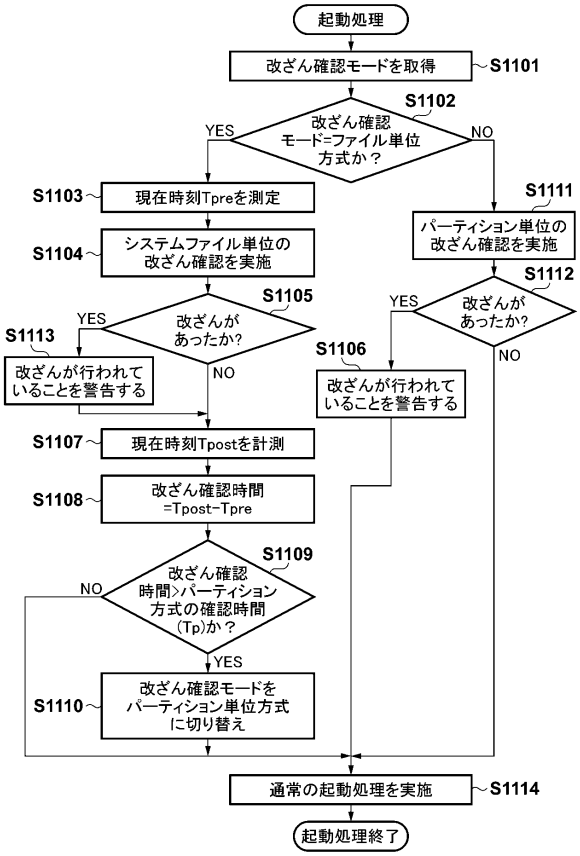
【図９】


ソフトウェア更新部起動
システムファイルパーティションをRWマウントする
S901
ファームウェアをダウンロード
S902
ダウンロードしたファームウェアの署名を検証
S903
S904
正しく検証ができたか？
NO
YES
アップデートフラグに署名を付けてフラグ領域に格納
S905
再起動を指示
S906
ソフトウェア更新部終了

【図１０】


起動処理
アップデートフラグの読み込み
S1001
アップデートフラグの署名検証
S1002
S1003
アップデート指示されたand署名検証成功?
NO
YES
アップデートを実施
S1004
S1005
正しくアップデートできたか?
NO
YES
システムファイルパーティションをROでリマウントする
S1006
システムファイルパーティションを読みだす
S1007
読みだしたシステムファイルパーティションのハッシュ値を算出
S1008
ハッシュ値を暗号化
S1009
暗号化したハッシュ値を保管する
S1010
通常の起動処理
S1011
起動完了

【図１１】


起動処理
改ざん確認モードを取得
S1101
S1102
改ざん確認
モード=ファイル単位
方式か？
YES
NO
S1103
現在時刻Tpreを測定
S1104
システムファイル単位の
改ざん確認を実施
S1105
改ざんが
あったか?
YES
NO
S1113
改ざんが行われて
いることを警告する
S1107
現在時刻Tpostを計測
S1108
改ざん確認時間
=Tpost-Tpre
S1109
改ざん確認
時間>パーティション
方式の確認時間
(Tp)か？
NO
YES
S1110
改ざん確認モードを
パーティション単位方式
に切り替え
S1111
パーティション単位の
改ざん確認を実施
S1112
改ざんが
あったか?
YES
NO
S1106
改ざんが行われて
いることを警告する
通常の起動処理を実施
S1114
起動処理終了

フロントページの続き

(72)発明者　土樋　直基
東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

審査官　上島　拓也

(56)参考文献　米国特許出願公開第２０１０／００６２８４４（ＵＳ，Ａ１）
特開２００８－２２６１５９（ＪＰ，Ａ）
特開２００９－２９４８５９（ＪＰ，Ａ）
特開２００９－１９３５２８（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　２１／６４
Ｂ４１Ｊ　２９／００
Ｂ４１Ｊ　２９／３８
Ｈ０４Ｌ　９／３２
Ｈ０４Ｎ　１／００