(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5646631号
(P5646631)
(45)発行日　平成26年12月24日(2014.12.24)　(24)登録日　平成26年11月14日(2014.11.14)

(51)Int.Cl.　F I
G06F 21/56 (2013.01)　G06F 21/00 156A
G06F 21/00 156G

請求項の数 39　(全 45 頁)

(21)出願番号	特願2012-525523 (P2012-525523)
(86)(22)出願日	平成22年8月11日 (2010.8.11)
(65)公表番号	特表2013-502639 (P2013-502639A)
(43)公表日	平成25年1月24日 (2013.1.24)
(86)国際出願番号	PCT/US2010/002214
(87)国際公開番号	WO2011/022040
(87)国際公開日	平成23年2月24日 (2011.2.24)
審査請求日	平成25年2月25日 (2013.2.25)
(31)優先権主張番号	61/286,369
(32)優先日	平成21年12月14日 (2009.12.14)
(33)優先権主張国	米国 (US)
(31)優先権主張番号	12/714,547
(32)優先日	平成22年3月1日 (2010.3.1)
(33)優先権主張国	米国 (US)

(73)特許権者　512038595
ファットスカンク・インコーポレーテッド
ＦＡＴＳＫＵＮＫ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国　カリフォルニア州９４０４１　マウンテン・ビュー，マリポサ・アベニュー，５９０
(74)代理人　110000028
特許業務法人明成国際特許事務所
(72)発明者　ヤコブソン・ビョルン・マルクス
アメリカ合衆国　カリフォルニア州９４０４１　マウンテン・ビュー，マリポサ・アベニュー，５９０

最終頁に続く

(54)【発明の名称】デバイスの監査

(57)【特許請求の範囲】
【請求項１】
物理メモリと、
プロセッサと、を備えるシステムであって、
前記プロセッサは、
ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、前記１つまたは複数のハードウェア・パラメータは、前記物理メモリの容量と前記物理メモリの速度との少なくとも一方を含み、
前記物理メモリに対して、少なくとも一つの書き込み処理を含む変更のシーケンスを実行し、
前記メモリが不正なプログラムによる影響を受けていないことを、部分的に前記物理メモリの速度に基づいて、検証するために、前記変更のシーケンスと関連づけられた複数の結果をベリファイアに提供するように構成されている、システム。
【請求項２】
請求項１に記載のシステムであって、
前記プロセッサは、さらに、前記物理メモリに対する前記変更のシーケンスを初期化するための初期化情報を受け取るように構成されている、システム。
【請求項３】
請求項２に記載のシステムであって、
前記初期化情報はシード値を含む、システム。

【請求項４】
請求項２に記載のシステムであって、
前記初期化情報はステップ値を含む、システム。
【請求項５】
請求項２に記載のシステムであって、
前記初期化情報はプロキシから受け取られる、システム。
【請求項６】
請求項２に記載のシステムであって、
前記初期化情報は外部ベリファイアから受け取られる、システム。
【請求項７】
請求項２に記載のシステムであって、
前記初期化情報は現象の表現である、システム。
【請求項８】
請求項７に記載のシステムであって、
前記初期化情報は、前記システムが備える加速度計から受け取られる、システム。
【請求項９】
請求項７に記載のシステムであって、
前記プロセッサは、少なくとも部分的に、現象を測定することにより、前記初期化情報を受け取るように構成されている、システム。
【請求項１０】
請求項１に記載のシステムであって、
前記プロセッサは、さらに、前記物理メモリのうち前記変更のシーケンスが実行される部分を関数に従って決定するように構成されている、システム。
【請求項１１】
請求項１に記載のシステムであって、
前記変更のシーケンスは、関数に従って実行される、システム。
【請求項１２】
請求項１に記載のシステムであって、
前記プロセッサは、
前記物理メモリの一部を選択的に読み取るように構成される、システム。
【請求項１３】
請求項１２に記載のシステムであって、
前記プロセッサは、前記物理メモリの前記選択的読み取りごとに増加される値に少なくとも部分的に基づいて定められるアクセス位置の前記物理メモリを選択的に読み取るように構成されている、システム。
【請求項１４】
請求項１３に記載のシステムであって、
前記増加される値の増加量は、前記物理メモリのサイズの測定値に対して、互いに素である、システム。
【請求項１５】
請求項１に記載のシステムであって、
前記プロセッサは、恒等関数ではない関数であって、前記物理メモリの一部を入力として受け取る関数を少なくとも部分的に用いることにより、前記複数の結果を決定するように構成されている、システム。
【請求項１６】
請求項１に記載のシステムであって、
前記プロセッサは、アキュムレータを少なくとも部分的に用いてメモリの内容をレジスタに累積することにより、前記結果を決定するように構成されている、システム。
【請求項１７】
請求項１６に記載のシステムであって、

前記アキュムレータは非線形関数を含んでいる、システム。
【請求項１８】
請求項１６に記載のシステムであって、
前記アキュムレータは、ＸＯＲおよびローテートを用いる、システム。
【請求項１９】
請求項１に記載のシステムであって、
前記プロセッサは、少なくとも部分的に、プロキシに情報を提供することにより、前記複数の結果を前記ベリファイアに提供するように構成されている、システム。
【請求項２０】
請求項１に記載のシステムであって、
前記ベリファイアは、物理メモリ変更の期待されるシーケンスが前記プロセッサにより実行されたかどうかを、受信した前記複数の結果に少なくとも部分的に基づいて判断するように構成されている、システム。
【請求項２１】
請求項１に記載のシステムであって、
前記判断は、前記変更のシーケンスの実行に関連するタイミング情報に少なくとも部分的に基づいている、システム。
【請求項２２】
請求項１に記載のシステムであって、
前記判断は、デバイスから受け取る結果の正確さに少なくとも部分的に基づいている、システム。
【請求項２３】
請求項１に記載のシステムであって、
前記ベリファイアは、デバイス上で前記プロセッサと併置されている、システム。
【請求項２４】
請求項２３に記載のシステムであって、
前記ベリファイアは、加入者識別モジュールに含まれている、システム。
【請求項２５】
請求項１に記載のシステムであって、
前記ベリファイアは、ネットワーク接続を介して前記プロセッサと通信する、システム。
【請求項２６】
請求項１に記載のシステムであって、
前記プロセッサは、さらに、前記ベリファイアからシードを取得するように構成されている、システム。
【請求項２７】
請求項１に記載のシステムであって、
前記変更のシーケンスは、前記ベリファイアから取得される入力に少なくとも部分的に依存する、システム。
【請求項２８】
請求項１に記載のシステムであって、
前記物理メモリ内にアクティブである回避的ソフトウェアはないと判断されたら、前記プロセッサは、スキャンを実行するように構成されている、システム。
【請求項２９】
請求項２８に記載のシステムであって、
前記スキャンは、ライセンスを受けていないソフトウェアのスキャンを含む、システム。
【請求項３０】
請求項１に記載のシステムであって、
前記メモリは秘密データを含み、

前記ベリファイアに結果を提供することによって、前記秘密データが漏洩することはない、システム。
【請求項３１】
請求項１に記載のシステムであって、
前記プロセッサは、携帯電話機の中に含まれている、システム。
【請求項３２】
ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取ることであって、前記１つまたは複数のハードウェア・パラメータは、物理メモリの容量と前記物理メモリの速度との少なくとも一方を含む、ハードウェア・パラメータを受け取ることと、
前記物理メモリに対して、少なくとも一つの書き込み処理を含む変更のシーケンスを実行することと、
前記メモリが不正なプログラムによる影響を受けていないことを、部分的に前記物理メモリの速度に基づいて、検証するために、前記変更のシーケンスと関連づけられた複数の結果をベリファイアに提供することと、を含む方法。
【請求項３３】
請求項３２に記載の方法であって、さらに、
初期化情報を受け取ることを含む、方法。
【請求項３４】
請求項３２に記載の方法であって、さらに、
前記物理メモリの一部を選択的に読み取ることを含む、方法。
【請求項３５】
コンピュータ読み取り可能な記憶媒体において実現されるコンピュータプログラムであって、
ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、前記１つまたは複数のハードウェア・パラメータは、物理メモリの容量と前記物理メモリの速度との少なくとも一方を含み、
前記物理メモリに対して、少なくとも一つの書き込み処理を含む変更のシーケンスを実行し、
前記メモリが不正なプログラムによる影響を受けていないことを、部分的に前記物理メモリの速度に基づいて、検証するために、前記変更のシーケンスと関連づけられた複数の結果をベリファイアに提供する、
ためのコンピュータ命令を含む、コンピュータプログラム。
【請求項３６】
請求項３５に記載のコンピュータプログラムであって、さらに、
初期化情報を受け取ることを含む、コンピュータプログラム。
【請求項３７】
請求項３５に記載のコンピュータプログラムであって、さらに、
前記物理メモリの一部を選択的に読み取ることを含む、コンピュータプログラム。
【請求項３８】
プロセッサと、
前記プロセッサに結合され、前記プロセッサに命令を提供するように構成されたメモリと、を備えるシステムであって、
前記プロセッサは、
デバイス上の物理メモリへの変更のシーケンスの前記デバイスによる実行に関連する１つまたは複数の結果を、前記デバイスから受け取り、
前記物理メモリの速度を含む、ハードウェア構成に対応する１以上のパラメータに部分的に基づいて、前記デバイス上の前記メモリ内には回避的ソフトウェアがないことを、前記１つまたは複数の結果が示していることを確認し、
前記確認が行われた後に、前記デバイスのスキャンを開始する、ように構成されている

、システム。
【請求項３９】
プロセッサと、
前記プロセッサに結合され、前記プロセッサに命令を提供するように構成されたメモリと、を備えるシステムであって、
前記プロセッサは、
少なくとも一つの書き込み処理を含む物理メモリへの変更のシーケンスと関連づけられた結果をデバイスが決定するように要求を開始させ、
前記デバイスから前記結果を受け取り、
前記結果と、前記結果を前記デバイスが提供するのに要した時間の長さと、前記物理メモリの速度と、に少なくとも部分的に基づいて、前記デバイスのセキュリティ状態を確認する、ように構成されている、システム。
【発明の詳細な説明】
【技術分野】
【０００１】
他の出願の相互参照
本出願は、“ＤＥＴＥＣＴＩＯＮ ＯＦ ＭＡＬＷＡＲＥ（マルウェアの検出）”という名称で２００９年８月１７日に出願された米国仮特許出願番号６１／２３４，６０４、“ＡＵＤＩＴＩＮＧ Ａ ＤＥＶＩＣＥ（デバイスの監査）”という名称で２００９年１１月２日に出願された米国仮特許出願番号６１／２５７，０４３、および“ＡＵＤＩＴＩＮＧ Ａ ＤＥＶＩＣＥ”という名称で２００９年１２月１４日に出願された米国仮特許出願番号６１／２８６，３６９の優先権を主張するものであり、これらの各々は、その全体が全ての目的のために参照により本明細書に組み込まれる。
【背景技術】
【０００２】
不正なプログラムの存在を検出するための既存の技術は、一般的に多くの資源を必要とする。例えば、概して、それらは（例えば、ブラックリストの）常時更新を必要とし、問題の定期的または継続的スキャンを必要とする。そのような技術により保護されるデバイスが、例えばメモリが限られている、あるいはバッテリを電源とするなど、資源に制限がある場合には、状況はさらに悪くなる。一例として、限られた資源をもつデバイスは、全ての既知の不正プログラムを検出するための定義を記憶することができないことがある。別の例として、不正プログラムをスキャンする動作は一般に多くの電力を必要とし、バッテリ駆動デバイスのバッテリをすぐに使い果たしてしまう恐れがある。環境によっては、不正プログラムの発見を容易にするために中央権限が用いられる。このアプローチの欠点の１つは、一般的に、保護されるデバイスが、デバイス・アクティビティの詳細なログを作成する必要があるということである。そのようなログの生成は、（例えば、大容量のディスク記憶、ログデータを編成するための処理能力、ログデータを中央局に伝送するための帯域幅を必要とするなど）多くの資源を必要とし、また、プライバシーの問題を生じる可能性がある。
【０００３】
また、不正なプログラムの存在を検出するための既存の技術は、概して、誤ったレポートの原因となるようなプログラムによる攻撃に対して、脆弱である。例えば、ルートキットは、アプリケーションによるオペレーティング・システムへの要求を“盗聴”することができ、これらの要求およびその応答に変更を加えることがある。どのようなプロセスが実行中であるかについての情報をアプリケーションが要求した場合に、悪質なルートキット・アプリケーションは、自身に関する情報を、オペレーティング・システムにより返されるレポートから削除することによって、検出を回避することができる。
【０００４】
不正プログラムのインストールまたは実行に対するスクリーニングのための既存の技術もまた、マルウェアの新しいインスタンスに対して脆弱であることが知られており、それ

らの構造および機能に関する情報が不足していることによって、直ちには検出できない場合がある。従って、デバイスで使用可能な資源にかかわりなく、不正プログラムが十分に精巧であり、さらに／またはこれまでに検出されたことがないものである場合は、それは検出を逃れて、検出されない害を引き起こすことがある。さらに、不正プログラムが、（例えば、ソフトウェアの違法コピーを容易にするため）ユーザによって意図的に検出を回避するようにインストールされた場合は、従来の技術では、不正プログラムまたはその他の不正なアクティビティを見つけられない場合がある。

【発明の概要】

【課題を解決するための手段】

【０００５】

本発明の様々な実施形態は、以下の詳細な説明および添付の図面において開示される。

【図面の簡単な説明】

【０００６】

【図１】図１は、デバイスの監査が行われる環境の一実施形態を示している。

【０００７】

【図２】図２は、デバイスの一実施形態を示している。

【０００８】

【図３】図３は、デバイスの監査を実行するプロセスの一実施形態を示している。

【０００９】

【図４】図４は、デバイスの監査を実行するプロセスの一実施形態を示している。

【００１０】

【図５Ａ】図５Ａは、図３に示すプロセスの実行前のメモリを表したものである。

【００１１】

【図５Ｂ】図５Ｂは、図３に示すプロセスが実行されている間のメモリを表したものである。

【００１２】

【図６】図６は、デバイスの監査を実行するプロセスの一実施形態を示している。

【００１３】

【図７】図７は、デバイスの監査に関連して用いられる擬似コードの一例を示している。

【００１４】

【図８】図８は、デバイスの監査を実行するプロセスの一例を示している。

【００１５】

【図９】図９は、デバイスの監査が行われる環境の一実施形態を示している。

【００１６】

【図１０】図１０は、デバイスの一部分の一実施形態を示している。

【００１７】

【図１１】図１１は、デバイスの監査を実行するプロセスの一実施形態を示している。

【００１８】

【図１２】図１２は、ｓｔｅｐによって読み取られるメモリの一部を示している。

【００１９】

【図１３】図１３は、選択的にメモリを読み取るプロセスの実装の一実施形態を示している。

【００２０】

【図１４】図１４は、デバイスの監査の一部における時間計測のプロセスの実装の一実施形態を示している。

【発明を実施するための形態】

【００２１】

本発明は、数多くの方法で実施することが可能であり、それには、プロセス、装置、システム、構成物、コンピュータ読み取り可能な記憶媒体上で実現されるコンピュータプログラム・プロダクトならびに／または、プロセッサに結合されたメモリに記憶された命令

および／もしくはそれにより提供される命令を実行するように構成されたプロセッサなどのプロセッサとしての実施が含まれる。本明細書では、このような実施の形態、または本発明が取り得るその他の形態を、技術と呼ぶ場合がある。一般的に、開示されるプロセスのステップの順序は、発明の範囲内で変更することができる。特に明記しない限り、タスクを実行するために構成されたものとして記載されるプロセッサまたはメモリなどのコンポーネントは、所与の時間に一時的にそのタスクを実行するために構成された汎用コンポーネント、またはそのタスクを実行するために作製された専用コンポーネントとして実装してもよい。本明細書において用いられる場合の‘プロセッサ’という用語は、コンピュータプログラム命令などのデータを処理するように構成された１つまたは複数のデバイス、回路、および／または処理コアを指している。

【００２２】

本発明の１つまたは複数の実施形態についての詳細な説明が、以下で、発明の原理を示す添付図面と共に提供される。本発明は、それらの実施形態に関連させて説明されるが、本発明は、いずれの実施形態にも限定されるものではない。本発明の範囲は、請求項によってのみ限定されるものであり、本発明は、多くの代替案、変形、および均等物を包含している。発明についての完全な理解を与えるため、様々な具体的詳細が以下の説明において記載される。これらの詳細は、例示目的で提供されるものであり、本発明は、これら特定の詳細の一部または全部を省いて、請求項に基づき実施することができる。明確にする目的で、発明が不必要に不明瞭になることがないよう、本発明に関連する技術分野で知られている技術的事項については詳細に記載していない。

【００２３】

図１は、デバイスの監査が行われる環境の一実施形態を示している。図示の例では、デバイス１０２は携帯電話機である。デバイス１０２は、（例えば、ネットワーク１０４を介して）ベリファイア１０６と通信する。図１では、デバイス１０２は、３Ｇネットワークを介してベリファイア１０６と通信する。ベリファイア１０６は、デバイス１０２への電話サービスの提供事業者といったキャリアの管理下にある。ベリファイア１０６は、デバイス１０２に対応するエントリとデバイス１０２上のＲＡＭ容量とを含むハードウェア構成情報のデータベースを有している。

【００２４】

以下でより詳細に説明するように、デバイス１０２を監査することが可能であり、これにより、デバイス上に存在する回避的プログラム（例えば、マルウェア）を検出および／または削除することができる。一部の実施形態では、これは、デバイス１０２上の物理メモリに対して一連の変更を実施することにより達成される。メモリ変更の実行に伴う結果は、ベリファイア１０６により検証される。デバイス１０２が、そのような回避的プログラムの影響を受けていないと判断されたら、追加のスキャンを実行することができる。これについても以下でより詳細に説明する。例えば、マルウェア（例えば、ユーザによる認知および／または同意なしにインストールされたソフトウェア）の検出に加えて、本明細書に記載の技術は、キャリアまたはハードウェア・メーカによりインストールされたデジタル著作権管理を回避するなどの目的で、ユーザにより行われた“ジェイルブレイク”処置（例えば、権限昇格）を検出することができる。

【００２５】

多様なデバイスを、本明細書に記載の技術と組み合わせて用いることができる。例えば、一部の実施形態では、デバイス１０２はビデオゲーム機である。ビデオゲーム機は、インターネット（１０４）を介して、ゲーム機メーカの管理下にあるベリファイアと通信するように構成されている。デバイス１０２の所有者が、（例えば、ＭＯＤチップを使用することにより）デバイス１０２への不正な変更を行うと、それによる変更をベリファイア１０６は検出することができる。

【００２６】

本明細書に記載の技術と組み合わせて用いることができるデバイスの他の例には、デスクトップ・コンピュータ、ノート型コンピュータ、ネットブック・コンピュータ、パーソ

ナル・デジタル・アシスタント、ビデオ再生装置（例えば、テレビ、ＤＶＤプレーヤ、ポータブル・ビデオプレーヤ）、ルータ、アクセスポイント、セットトップボックス、医療機器、さらにはプロセッサとメモリとを有するその他のほぼ全てのデバイスが含まれる。

【００２７】

様々な実施形態において、ベリファイア１０６は、別個の機関(entity)による代わりに、デバイス１０２のユーザによって制御される。例えば、デバイス１０２のユーザが所有するデスクトップ・コンピュータを、デバイス１０２に検証サービスを提供するように構成することができる。この場合、デバイス１０２は、ローカル・ネットワークを介してベリファイアと通信するように構成することができる。また、デバイス１０２は、（例えば、専用ケーブルによって）ベリファイア１０６と直接通信するように構成することもでき、ネットワーク１０４は可能であれば省略される。

【００２８】

一部の実施形態では、ベリファイアは、デバイス１０２と併置されるか、あるいは直接それに接続されている。例えば、携帯電話機に差し込まれた加入者識別モジュール（“ＳＩＭ”）カードを、ベリファイア１０６の機能を携帯電話機に提供するように構成することができる。別の例として、ベリファイア１０６の機能を、携帯電話機の充電に使用される電源コードに組み込むことができる。これらの実施形態では、外部のベリファイアは、省略することができるか、または併置／接続されたベリファイアにより提供される検証サービスに追加して用いることができるか、いずれかである。一例として、デバイス１０２は、内蔵のＷｉ‐Ｆｉ機能を備えるパーソナル・ビデオプレーヤである。デバイスの充電に使用される電源コードを、充電のたびにデバイスに検証サービスを提供するように構成することができる。加えて、Ｗｉ‐Ｆｉ無線機がアクティブであれば、デバイスは、デバイスのメーカにより提供されるベリファイアと定期的に通信するように構成することができる。別の例として、ベリファイア１０６は、ユーザにより定期的にラップトップ１０２に差し込まれるＵＳＢデバイスに組み込むことができる。加えて、ラップトップ１０２のユーザがオンライン銀行と銀行取引を行おうとするたびに、銀行が、ユーザのアカウントへのアクセスを許可する前に、ラップトップ１０２に検証サービスをさらに提供することができる。さらに別の例として、ユーザがネットワークへの接続またはサービスへのアクセスを許可される前に、ネットワーク・オペレータまたはサービス・プロバイダが、ユーザに自身のマシンを監査してもらうように要求することができる。また、ユーザが、例えば危険そうな状況にさらされたことに気付いたら、その後に監査を起動することもできる。ユーザが監査を起動することが可能な一つの方法は、デバイス上でメニュー・オプションを選択することである。別の方法例は、ユーザが、（例えば、ウェブ・フォームによってオンライン要求を送信することにより）ベリファイア１０６による監査を要求することである。

【００２９】

図２は、デバイスの一実施形態を示している。図示の例では、デバイス１０２は、プロセッサ２０２と、第１のメモリ２０４と、第２のメモリ２０６と、通信インタフェース２０８とを備えている。一例として、デバイス１０２は、５２８ＭＨｚのＡＲＭプロセッサ（２０２）と、１２８ＭＢのＲＡＭ（２０４）と、ユーザにより１ＧＢのマイクロＳＤカード（２０６）が挿入されたマイクロＳＤカード・スロットと、３Ｇモデム（２０８）とを備えている。メモリ２０４は、本明細書において“高速”メモリとも呼ばれる。メモリ２０６は、本明細書において“低速”メモリとも呼ばれる。しかしながら、メモリ２０４および２０６は、異なる速度である必要はない。さらに、ＧＰＳ受信機（図示せず）など他の構成要素を、デバイス１０２内に備えていてもよい。第２のメモリ２０６などの要素は、可能であれば省略してもよい。アクティブなプログラムを格納することが可能なＲＡＭを高速と呼ぶことができ、データのみ記憶することが可能なＲＡＭを低速とみなすことができる。

【００３０】

本明細書に記載の監査技術を用いて、高速メモリ内にアクティブなプロセスが無いこと

を確認することができる。そして、この確認が完了した後に、全てのメモリ（例えば、高速と低速の両方）をスキャンすることが可能であり、これによって、高速および低速のメモリの内容またはその一部を識別し、分類し、レポートし、場合によっては変更を加える。高速と低速のメモリは様々に区別することができる。例えば、ＲＡＭ、フラッシュメモリ、およびハードディスク・ドライブを備えるデバイスでは、ＲＡＭのみを高速メモリとし、フラッシュメモリとハードディスク・ドライブを低速メモリとして扱うことが可能である。また、ＲＡＭとフラッシュメモリの両方を高速メモリとし、ハードディスク・ドライブを低速メモリとして扱うことも可能である。さらには、所与のデバイス上に物理的に配置されている全てのメモリを高速であるとし、そのデバイスによりアクセス可能な（またはアクセス可能になる可能性がある）外部メモリを低速とみなすこともできる。外部コンポーネントと通信するためのターンアラウンド時間が原因となって、そのような外部アクセスは、外部メモリのタイプおよび実際のローカルアクセス速度にかかわりなく、より低速となる。どのようなタイプのメモリを高速／低速として扱うかにより、それに応じてパラメータの選択が行われることになる。

【００３１】

以下でより詳細に説明するように、デバイス１０２に不正な変更が加えられていることは、メモリ２０４に対して一連の変更を実施し、その結果を調べるようにデバイス１０２を構成することにより検出することができる。例えば、変更を実行するのにかかる時間が所定の時間長の許容誤差を超える場合、または変更に伴って決定された結果が期待される結果と一致しない場合、これは、回避的プログラムの存在を示し得る。様々な実施形態において、メモリへの変更は、メモリ２０４といった高速メモリに対してのみ実行されるのではなく、デバイス上の全てのメモリ（例えば、メモリ２０４とメモリ２０６の両方）に実行される。

【００３２】

図３は、デバイスの監査を実行するプロセスの一実施形態を示している。様々な実施形態において、図３に示すプロセスはデバイス１０２により実行される。図３に示すプロセスは、様々な方法で起動することができる。（例えば、電力供給を検出したらプロセスを起動するようにデバイスを構成することにより）例えばユーザがデバイスを充電するたびに、プロセスを起動することができる。また、特に大量のトランザクションまたは普段と異なるトランザクションの発生に応じて；ユーザが危険にさらされているという懸念に応じて（例えば、極めて悪質な個人により新しい脆弱性がリリースされたという通知をキャリアが受けたことに応じて）；一定の時間が経過したことなどに応じて、プロセスを起動することもできる。図３に示すプロセスの起動をトリガすることができる事象のさらなる例には、支払いまたはその他の金融取引をしようとするデバイス１０２のユーザによる試行、（例えば、デバイスのユーザが銀行口座にアクセスしようとする）認証試行、およびアクセス要求（例えば、デバイスへの映画のダウンロードの要求）の実行が含まれる。

【００３３】

ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取る３０２で、プロセスは開始する。ハードウェア・パラメータの例には、高速メモリ２０４の容量および速度が含まれる。例えば、図２に示すデバイスの場合、ハードウェア・パラメータは、“容量＝１２８Ｍ”および“速度＝３００ＭＨｚ”を含むことになる。使用することができる追加のパラメータには、コアの数、バスのタイプなどが含まれる。

【００３４】

ハードウェア・パラメータは、様々な方法で受け取ることができる。一例として、携帯電話機のＳＩＭが、搭載されているメモリの容量および速度を検出するように構成することができる。別の例として、デバイス１０２を電源に（またはコンピュータまたは他のデバイスに）接続するのに専用のケーブルを使用する場合には、特定の容量および速度のメモリを有するデバイスとの組み合わせでのみ機能するケーブルによって、パラメータを知る（そして、このようにして“受け取る”）ようにしてもよい。さらに別の例として、デバイスのシリアル番号が、デバイスに搭載されている高速メモリ２０４の容量および速度

を示していてもよい。様々な実施形態において、ユーザ（またはその代表者）は、メモリのパラメータをウェブ・フォームまたは構成ファイルに入力するように要求される。また、可能性が高いデバイスのメモリ構成について仮定することもでき、ベンチマーク・プログラムを実行することにより、その仮定が正しい可能性が高いかどうか確認することができる。

【００３５】

３０４において、物理メモリに対して変更のシーケンスが実行される。そのような変更を行うことができる方法の例について、以下でより詳細に説明する。一部の実施形態では、実行される変更のシーケンスは、ベリファイアによって決定される。行われるべき変更のセットは、様々な方法でデバイスに提供することができる。例えば、シーケンスは、デバイス上でシード値に基づいて構築することができる。シーケンスは、製造時に、またはサプライヤもしくはキャリアへの出荷時に、または購買時に、デバイス上にプリロードすることができる。また、ユーザ選択により、またはサービス・プロバイダにより、購入後の任意の時点で（例えば、上書き更新として、もしくはファームウェアの更新として）、または監査の実行が必要なときに、ロードすることもできる。パラメータ化は、仕様が分かっていることを前提として、メーカまたはサプライヤまたはキャリアによって行うことができる。また、それを、ユーザまたはサービス・プロバイダによって、例えばシリアル番号のルックアップにより行うこともできる。パラメータは、モデルまたはデバイス名に関連付けることができる。デバイスが、例えばコンポーネントの置換または追加により再構成される場合は、それらの新しいコンポーネントにより、新規または追加のパラメータ化に関する情報を伝えることができる。それらのコンポーネントは、パラメータだけではなく、命令のセット全体を伝えることもできる。あるいは、コンポーネントのシリアル番号、名前、またはタイプによって、パラメータに必要な変更を示すことができる。クライアント・デバイスが、アルゴリズムまたは新しいコンポーネントのインストール時に安全であると考えられる場合には、クライアント・マシンは、（システムの起動の際に一般的に行われているように）どのようなコンポーネントがインストールされたのか問い合わせて、それに応じてパラメータを設定することもできる。

【００３６】

様々な実施形態において、デバイス・メーカは、有効化されていない監査ソフトウェアを無料でプリロードをすることを提案し、後に、監査サービスおよび／または以下でより詳細に説明する追加のスキャン・サービスを有効化するための支払いを要求する。その後、監査ソフトウェアを、エンドユーザまたはサービス・プロバイダによる要求に応じて、キャリアによって有効化することができる。キャリアは、有効化の料金を徴収し、そして任意に、その料金の一部を、携帯電話機メーカ、監査ソフトウェアのプロバイダ、追加のスキャン・ソフトウェア（例えば、アンチウイルス検出サービス）のプロバイダ、および他の取引関係機関に送る。

【００３７】

３０６において、３０４で実行されるプロセス部分による１つまたは複数の結果が、ベリファイアに報告される。一部の実施形態では、結果はプロキシ９０６に提供され、これにより、結果はタイムスタンプが付されて、ベリファイアに提供される。図５に関連して説明するように、一部の実施形態では、メモリへの変更と、ベリファイアとの通信が、複数回繰り返され、これに応じて、図３および４に示すプロセスを適応させる。

【００３８】

図４は、デバイスの監査を実行するプロセスの一実施形態を示している。様々な実施形態において、図４に示すプロセスは、ベリファイア１０６により実行される。上記で説明したように、一部の実施形態では、図４に示すプロセスは、（キャリアにより管理されるベリファイア上など）デバイス１０２とは別の機関によって実行される。他の実施形態では、このプロセスは、デバイス１０２上に配置されているか、またはそれに物理的に接続されているベリファイアによって実行される。

【００３９】

４０２において、結果を受け取ると、プロセスが開始する。例えば、デバイス１０２が３０６で結果を報告すると、これらの結果が４０２でベリファイアにより受け取られる。

【００４０】

４０４で受信した結果が、物理的変更の期待されるシーケンスが行われたことを示しているか否か、４０４において判断される。ベリファイア１０６は、（不正な変更が加えられていないと仮定した場合に）デバイス１０６上でのメモリ変更のシーケンスの実行にかかるはずの時間などの情報を持つように構成されている。一部の実施形態では、ベリファイア１０６は、さらに、シード値、およびデバイス１０２によって実行された計算の結果など、追加情報を記憶するように構成されている。

【００４１】

物理メモリ変更の期待されるシーケンスが行われた（例えば、デバイス１０６がメモリ変更のシーケンスを実行した）と判断される場合、デバイスに不正な変更が加えられていないと結論される（４０６）。また、以前にデバイス１０２上でアクティブであった可能性のある回避的プロセスは中和されている。（例えば、シーケンスの実行にかかった時間が基準からはずれている、または計算結果が正しくないという理由で）物理メモリ変更の期待されるシーケンスが行われていないと判断される場合は、デバイスに不正な変更が加えられている（例えば、回避的プロセスがデバイス上に存在して、しかも検出を回避しようとしている）と結論される（４０６）。様々な実施形態において、ネットワーク・ノイズに起因する誤りを回避するために、誤り訂正符号が用いられる。コンテンツのアクティブな改ざんを回避するために、メッセージ認証符号および他の認証技術を用いることができる。コンテンツを難読化して、伝送される平文メッセージを盗聴者が確認できないようにするために、暗号化技術を用いることができる。

【００４２】

図５Ａは、図３に示すプロセスの実行前のメモリを表している。図示の例では、カーネル５０２、許可プログラム５０４、不正プログラム（例えば、マルウェア・エージェント）５０８、および監査プログラム５０６がＲＡＭにロードされている。一般に、回避的プログラムが、デバイス上に常駐し続けるためには、２つのことのうち１つを行う必要がある。ＲＡＭ（またはスワップ領域）においてアクティブであり続けるか、あるいは、スキャンが実行された後にマルウェア・エージェントによる掌握が可能となるように、正規のプログラム、データ、またはデバイスの構成に変更を加えるか、いずれかが必要である。以下でより詳細に説明するように、本明細書に記載の技術を用いることで、マルウェア・エージェントの存在を、それが採用している検出回避技術にかかわりなく検出することができる。さらに、本明細書に記載の技術を用いると、監査プログラム５０６がマルウェア・エージェント５０４の後にロードされる場合でも、マルウェア・エージェントの存在を検出することができる。

【００４３】

図５Ｂは、図３に示すプロセスが実行されている間のメモリを表している。以下でより詳細に説明するように、監査部５０６は、監査部５０６により使用されている領域を除いて、ＲＡＭ（およびスワップ領域）の記憶をクリアするように構成されている。様々な実施形態において、さらに、他のサービスの最小限のセットによるＲＡＭの占有が許される。例えば、デバイス１０２が３Ｇ通信をサポートしている場合、監査部５０６がベリファイア１０６との通信に３Ｇモデムを使用することができるように、３Ｇドライバ／モジュールにより占有されているＲＡＭ領域はクリアされない。別の例として、一部の実施形態では、マイクロカーネルはＲＡＭの一部を占有することが許可されて、監査部５０６はＲＡＭのその残りの部分をクリアする。

【００４４】

図６は、デバイスを監査するプロセスの一実施形態を示している。デバイス１０２などのデバイス上で実行される監査プロセスにより、監査プロセスによる使用が要求されていないメモリ２０４（およびスワップ領域）の全ての部分がクリアされる６０２で、プロセスは開始する。一部の実施形態では、これには、カーネル、種々のドライバ、および他の

全てのプロセスのアンロードが含まれる。様々な実施形態において、要求のないメモリ領域は、（例えば、ゼロで）クリアされるのではなく、シーケンスによって上書きされる。シーケンスの一例は、擬似ランダム・シーケンスであり、これは、ＸＯＲ演算を用いることなどにより、元のメモリの内容と組み合わせられる。これによって、先に使用した擬似ランダム・シーケンスに相補的であるか、または等しい擬似ランダム・シーケンスと繰り返し組み合わせることにより、要求のないメモリ領域を後に再生することが可能となる。また、要求のないメモリ領域は、それをクリアするように、但しそのデバイスの標準的な消去操作には相当しない内容で、上書きすることもできる。例えば、０１０１０１０１というシーケンスまたは他の適当なシーケンスを書き込むことにより、要求のないメモリをクリアすることが可能である。

【００４５】

一部の実施形態において、監査コードは、ローダと可変アルゴリズム・セグメントという２つのコンポーネントを含んでいる。ローダの役目は、非ＲＡＭ記憶装置（例えば、メモリ２０４以外のもの）からアルゴリズム・セグメントをロードし、ロードされたアルゴリズム・セグメントに制御を引き渡すことである。アルゴリズム・セグメントが完了すると、その後ローダに制御が返される。

【００４６】

６０４において、メモリ２０４の内容がベリファイア１０６に報告される。一部の実施形態では、内容全体が報告される。他の実施形態では、前回の監査以降の変更についての記述のみが伝えられる。

【００４７】

６０６において、デバイスはベリファイアから暗号化シードを受け取る。シードは擬似ランダム・ストリングに展開され、そのストリングはＲＡＭに書き込まれる。プロセス６００の部分６０６に従ってＲＡＭにストリングを書き込む方法の例を以下で提示する。

【００４８】

６０８において、デバイスはベリファイアから暗号鍵を受け取る。

【００４９】

６１０において、デバイスは受け取った鍵を使用して、デバイスのＲＡＭの内容全体の鍵付きハッシュを計算する。

【００５０】

６１２において、デバイスは、結果として得られる値をベリファイアに報告する。ベリファイア１０６は、例えば図４に示すプロセスによって、その結果を評価する。

【００５１】

様々な実施形態において、デバイス１０２は、６０６および６１０での計算からの状態情報を、ベリファイア１０６により設定された時間間隔で報告する。そのような間隔を用いることによって、デバイス１０２により実行される計算が（例えば、メモリ２０６の一部ではなく）メモリ２０４内で実行されていることが保証される。

【００５２】

デバイス１０２は、必要に応じて、ベリファイア１０６からシードと各鍵の更新を得る。更新を用いることによって、デバイス１０２が計算を外部の高速資源にアウトソーシングしていないことが保証される。例えば、計算をアウトソーシングするためには、回避的プログラムは、シードと鍵の更新を外部デバイスに転送しなければならず、これによって測定可能な遅延が引き起こされることになる。

【００５３】

ベリファイア１０６は、最終的な関数値と途中結果の両方が正しく、また、許容期限内にデバイス１０２によってベリファイアに報告されたことを確認する。監査部がそのタスクを実行するのに要する時間を評価するための方法の例を以下で提示する。上述のように、一部の実施形態では、ベリファイア１０６はデバイス１０２の外部にあって、デバイスの所有者以外の機関により運営されている。他の実施形態では、ベリファイア１０６は、デバイス１０２のユーザの管理下にある。

【００５４】
図６に示すプロセスが完了した後、監査部５０６は、完全であるか部分的であるかはともかく、デバイスの内容を復元して、前のアクティブなプロセスまたはメモリ内容のさらなるスキャンを実行するプロセスに制御を返すことができる。高速メモリの内容は、それらが時間計測される計算の実行前に低速メモリにスワップアウトされていれば、あるいは元の内容がストリングと組み合わせられていれば、復元することができ、後者の場合は、同様の組み合わせを実行することにより以前の状態を復元することが可能である。また、“起動”状態をロードすることによりデバイスを再起動させることも可能である。さらには、メモリの内容をスキャン、レビュー、レポート、および変更するプロセス、またはこれらの操作の任意のサブセット（以下でより詳細に説明する）のプロセスに、最初に制御を引き渡すことが可能である。レポートは、ベリファイア１０６に提示することができ、またはメモリ内容の処理の管理を担当するものなど第三機関に提示することができる。後者の場合、ベリファイア１０６は、悪意のあるアクティブなプロセスが存在しないことを保証することを担当する場合があり、そして第２のベリファイアはデバイスのメモリの処理を担当することができ、これにより、マルウェアの検出、デジタル著作権管理に関係し得る特定のポリシー、またはデバイスのメモリのどのような内容が望ましいのかを識別する他のポリシーにそれが従うものであるかどうか判断される。
【００５５】
攻撃戦略の例
【００５６】
回避的プログラムが、例えば図６に示すプロセスの部分６０４での検出を回避するためには、それは、固有のプロセス（５０４）として、あるいは監査部５０６の破損版の一部として、ＲＡＭにおいてアクティブでなければならない。以下は、マルウェア・エージェント５０４などの回避的プログラムがアクティブであり続けようとすることができる方法の６つの例である。
【００５７】
戦略１：記憶装置をアウトソーシングする
【００５８】
監査部１０６が、（例えば、６０２で）適切な空間をクリアすることなく、さらに６０６で生成される擬似ランダム・ストリングの対応する部分を記憶するのに非ＲＡＭ記憶装置または外部記憶装置を頼るようにさせることにより、マルウェア・エージェントは、ＲＡＭにおいてアクティブであり続けて、検出されない状態を継続させようとすることができる。この場合、６１０での計算は、マルウェア・エージェントが常駐する空間の代わりに、アウトソーシングした記憶装置を使用するように変更されることになる。
【００５９】
戦略２：欠落データを計算する
【００６０】
擬似ランダム・ストリングの一部をアウトソーシングにより記憶する代わりに、マルウェア・エージェントは、ストリングの変形された表現（例えば、圧縮版、または欠落部分がある版）を記憶して、６１０における鍵付きハッシュの計算の際に、ストリングの関連部分を必要に応じて再構成することができる。マルウェア・エージェントは、擬似ランダム・ストリングを生成する元となるシードを有しているので、これ―またはその後の擬似乱数発生器の状態―を利用して、データの必要な部分を再生することができる。
【００６１】
戦略３：計算をアウトソーシングする
【００６２】
（Ｗｉ‐Ｆｉ接続など必要な通信基盤はやはり有効であると仮定して）マルウェア・エージェントは、関連データを外部デバイスに転送することができる。外部デバイスは、デバイス１０２からデータを受信して、ベリファイア１０６へのレポートに必要な値を計算し、それらの値をデバイス１０２上のマルウェア・エージェントに送る。

【００６３】
戦略４：検出コードに変更を加える
【００６４】
マルウェア・エージェントは、監査部５０６のコードを、変更を加えたコードで置き換えようとすることができる。この代替コードは、損なわれたメモリ内容についてのレポートを抑止するか、または監査の完了後にマルウェア・コードがロードされるためのフックを含むように設計されていてもよい。マルウェア・エージェントは、監査コードの一部をスワップアウトまたは圧縮し、それを必要に応じて再びロードまたは解凍することにより、空間をそれほど多くは占有することなく、そのような変更を監査部５０６に組み込むことを試みることができる。
【００６５】
高速メモリの充填
【００６６】
このセクションでは、図６に示すプロセスの部分６０６に関連して用いることができる技術の例について説明する。
【００６７】
図７は、デバイスの監査に関連して用いられる擬似コードの一例を示している。図示の例では、サブルーチンｇｅｔ＿ｐｅｒｍｕｔａｔｉｏｎが、０からｎｕｍｂｅｒ＿ｂｌｏｃｋｓ－１までのｎｕｍｂｅｒ＿ｂｌｏｃｋｓ個の要素のランダム順列を示すベクトルを返す。ここで、ｎｕｍｂｅｒ＿ｂｌｏｃｋｓは、ＲＡＭを構成するフラッシュメモリ・ブロックに等しいサイズ部分の数から、監査部が必要とするものを引いたものである。サブルーチンｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋは、擬似ランダムに生成されたビットのチャンクを返すものであり、チャンクという用語は、メモリバス上に伝送することができるデータ量を指して用いられる。一例として、Ａｎｄｒｏｉｄ Ｇ１電話機の場合、１チャンクは３２ビットである。
【００６８】
ｇｅｔ＿ｐｅｒｍｕｔａｔｉｏｎおよびｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋは、両方とも、最も新しく提供されたシードを入力として用いる。擬似ランダム・ストリングは、$segment_i \leftarrow hash(segment_{i-1})$として計算することができ、すなわち、ランダム・アクセスを用いて計算することができないように計算することができる。一例は、ハッシュ関数の非準同形特性を前提として、ハッシュ関数の繰り返し適用に基づく関数である。多様なハッシュ関数を用いることができる。一例は、５１２ビット・モードのＭＤ６である。
【００６９】
定数ｒｏｕｎｄｓは、関数ｍｏｄｉｆｙ＿ｍｅｍｏｒｙを用いて、擬似ランダム・チャンクがセルの内容にＸＯＲされる回数である。ｒｏｕｎｄｓの選択により、攻撃者が（欠落データを計算する）第２の攻撃戦略を実行するために行わなければならない仕事量を制御し、また、これが大きな値の場合にアルゴリズムの正当な実行にかかるコストを増加させもする。図示の例において、ｒｏｕｎｄｓ＝２とすると、各セルの値は２つの他のセルに依存するようになるので、ｒｏｕｎｄｓ＝１であるよりも、攻撃者に著しく大きなコストがかかることになる。これによって、攻撃者のメモリ管理戦略を混乱させることができる。図示の例において、ｃｈｕｎｋｓ＿ｐｅｒ＿ｂｌｏｃｋは、フラッシュメモリ・ブロックに含まれるチャンクの数であり、Ｇ１電話機の例では３２７６８（＝１２８ｋＢ／３２ビット）であり、このときｎｕｍｂｅｒ＿ｂｌｏｃｋｓ＝１０２４（＝１２８ＭＢ／１２８ｋＢ）である。
【００７０】
関数ｍｏｄｉｆｙ＿ｍｅｍｏｒｙ（ｐｏｓ，ｓｔｒｉｎｇ）は、位置ｐｏｓの内容を値ｓｔｒｉｎｇでＸＯＲするものであり、ｐｏｓ＝０は、演算されるのがＲＡＭの最初のチャンクであることを表しており、ｐｏｓ＝ｎｕｍｂｅｒ＿ｂｌｏｃｋｓ×ｃｈｕｎｋｓ＿ｐｅｒ＿ｂｌｏｃｋ－１は、最後のチャンクであることを表している。

【００７１】
ＲＡＭ（２０４）の代わりにフラッシュメモリ（例えば、メモリ２０６）を使用するように強制された場合、図７に関連して説明したメモリアクセス構造によって、ランダムに順序付けられたブロックの個々のページへのアクセスが生じる。これによって、フラッシュメモリは圧倒的な確率でクリアされることになり、また、擬似ランダム・アクセス順序によって、攻撃者がこの欠点を回避するようにメモリアクセスをスケジューリングすることを防止する。フラッシュメモリ内での計算のコストは、アルゴリズムの正当な実行に有効なＲＡＭ内での場合に比較して、著しく多くの時間を費やすものである。
【００７２】
一部の実施形態では、１つのハッシュ関数の適用を利用して、ｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋのいくつかの呼び出しを生成する。これによって、監査プロセスに伴う計算負荷が軽減され、このことは、タスクを実行する時間面でのメモリアクセスの貢献を際立たせている。
【００７３】
様々な実施形態において、ハッシュ関数への入力は、一定数の前の出力であり、このことは、擬似乱数発生器の特定の部分の状態を再構成することを望むマルウェア・エージェントによる記憶を複雑にしており、そのため、（欠落データを計算する）戦略２を利用しようとする試みをさらに妨げるために有効である。
【００７４】
時間計測の実行
【００７５】
このセクションでは、監査タスクの実行の時間計測に用いることができる技術の例について説明する。例えば、一部の実施形態では、図６に対応する記述部分に関連して述べたような技術が、ベリファイア１０６により採用される。
【００７６】
ベリファイア１０６は、例えば、記憶装置をアウトソーシングしようとする試み、欠落データを計算しようとする試み、および計算をアウトソーシングしようとする試みを識別するため、図６に示すプロセスの部分６０６および６１０の実行を時間計測するように構成されている。
【００７７】
一部の実施形態において、ベリファイア１０６は、（例えば、ベリファイア１０６により設定される）小刻みな間隔で、デバイス１０２から状態情報を取得するように構成されている。状態情報の一例は、最後に更新されたメモリ・チャンクのメモリ内容であり、これは、デバイス１０２がこの計算段階に達したことを証明するものである。ベリファイア１０６は、一定時間ごとに、デバイス１０２に更新要求を送信する。一部の実施形態では、更新要求は、サブルーチンｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋの出力の計算に用いられる擬似乱数発生器の状態の更新に対応している。サブルーチンｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋの出力が、既に生成された擬似ランダム・ストリングの未使用部分を選択することにより生成される場合、このストリングを同時にクリアすることができ、このようにして新しいシードを直ちに状態に影響させるように強制する。
【００７８】
攻撃戦略３（すなわち、計算のアウトソーシング）を採用する回避的プログラムは、計算を実行する外部デバイスに、擬似ランダム・ストリングの更新を送信する必要があり、その後に、外部デバイスは、デバイス１０２によってベリファイア１０６に報告されるべき次の結果値を計算し、これを回避プログラムに送信しなければならない。これによって、ラウンドトリップの遅延が生じる。ラウンドトリップの遅延が計時チェックポイント間の時間を超える場合は、不正行為が検出される。ここで、シードおよび鍵は、他の状態情報と共に、クライアント・デバイスとベリファイアとの間で安全に伝達されるものと仮定する。これを達成するために様々な暗号化技術を使用することができる。
【００７９】

様々な実施形態において、最悪な場合として輻輳のない環境を仮定して、デバイス１０２が備える通信機器（例えば、Ｗｉ‐Ｆｉ）を用いて計算をアウトソーシングするのに十分な時間がないように、デバイス固有のチェックポイント間の時間が選択される。

【００８０】

ｍｏｄｉｆｙ＿ｍｅｍｏｒｙの実行時間は、上記のパラメータの選択、および、どのようなハッシュ関数を用いてｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋを計算するのかによって、決定される。例えば、ＭＤ６ハッシュ関数は、２２４から５１２ビットまで様々に異なる出力サイズに設定することができる。上記で説明したように、一部の実施形態では、５１２ビット版が使用される。ｍｏｄｉｆｙ＿ｍｅｍｏｒｙの呼び出しごとの時間は、上記で決定されるチェックポイント間の時間よりも著しく小さい。

【００８１】

様々な回避的プログラムの検出例

【００８２】

以下のセクションでは、上述の様々な戦略を採用する回避的プログラムを、本明細書に記載の技術を用いて検出する方法の例を提示する。

【００８３】

記憶装置をアウトソーシングする－攻撃戦略１に対して防御する

【００８４】

空のＳＤカードがデバイス１０２に挿入されたと仮定する。これに対応する書き込み速度は５ＭＢ／ｓに達することがある。上述のように、ｍｏｄｉｆｙ＿ｍｅｍｏｒｙにより処理されるブロックの大きさは、この例では、１２８ｋＢに選択される。ＳＤカードにデータを書き込む時間は２５ｍｓとなる。これに対して、デバイス１０２上のＲＡＭは１００ＭＢ／ｓの書き込み速度であるとする。これに対応する書き込み時間は１．２５ｍｓとなる。追加される遅延は容易に検出することができる。また、２つのチェックポイント間でＳＤカードへの複数のアクセスが行われる場合は、追加される遅延の検出がさらに容易となる。

【００８５】

欠落データを計算する－攻撃戦略２に対して防御する

【００８６】

上述のように、擬似ランダム・ストリングは、ランダム・アクセスを用いて計算することができないように計算することが可能である。ある出力の値を計算するには、記憶しているデータから対応する入力を計算する必要がある。ｒｏｕｎｄｓ＞１であるので、ＲＡＭに記憶しているデータは、この必要な状態ではなく、２つ以上のｒｏｕｎｄｓの状態の組み合わせである。状態は、その符号空間の一部として、マルウェア・エージェントにより明示的に（ＲＡＭに）記憶される必要があり、これから必要な状態が計算される。このため、マルウェア・エージェントは、図６に示すプロセスの部分６１０の実行中に、“正規の”演算を行うために必要な計算に加えて、少なくともｒｏｕｎｄｓ×ｎｕｍｂｅｒ＿ｂｌｏｃｋｓ×ｃｈｕｎｋｓ＿ｐｅｒ＿ｂｌｏｃｋの（そして、実際には、それよりずっと多くの）ハッシュ演算を行うことを強いられる。それは、選択されたパラメータに対して、１億より多くのハッシュ演算となる。ハッシュ関数呼び出しを計算するためのおおよその時間が１０μｓであるとして、これは約１０００ｓであり、予想よりもおよそ３桁大きく、従って検出が可能である。

【００８７】

つぎに、戦略２に対する防御に対応する論証の実例を提示する。回避的プログラムはメモリ２０４内に常駐しており、それ自体とその変数のために、あるｃ個の３２ビット・チャンクの少なくとも一部を占めていると仮定する。最悪の場合として、変数の記憶のためにこの空間の全てを効果的に使用することが可能であると仮定することができる。これは、可能ではないが、マルウェアが検出されないままでいるために実行する必要のある仕事量の下限を与えるものである。実際には、ｃ個のチャンクを全て記憶のために用いることはできず、一部はそのコードを記憶する必要があるので、その労力はそれより大きい。

【００８８】
関数を計算するのに必要な値を保持していないｃ個のＲＡＭチャンクのヒットのそれぞれについて、マルウェア・エージェントは、期待される内容を計算しなければならない。この例では、ＲＡＭの充填が行われるより前の元の内容はゼロであったと仮定している。これがそうでない場合は、マルウェア・エージェントの労力はより大きくなり、従って、この仮定は、マルウェア・エージェントの労力についての下限を設定している。ＲＡＭ充填アルゴリズムにより行われた、このセルへの予想される更新を計算するため、マルウェア・エージェントは、当該メモリ・チャンクの値を全てのｒｏｕｎｄｓパスについて計算する必要がある。メモリにＸＯＲされる値は、擬似ランダム・シーケンスによるものである。そして、欠落しているセルのチェーンの状態を再構成することは、記憶のｃ個のチャンクの一部にマルウェア・エージェントにより記憶されている値から、値ｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋを計算することによってのみ可能である。変数はＲＡＭのみに記憶されていると想定されるか、またはマルウェア・エージェントはさらに（記憶装置をアウトソーシングする）戦略１も成功させる必要があると想定される。

【００８９】
上記で説明したように、擬似乱数発生器は、ランダム・アクセス・アプローチを用いて計算を行うことはできない。チャンク・サイズが３２ビット、状態のサイズ（＝ＭＤ６の出力サイズ）が５１２ビットであると、状態を記憶するためにＬ＝１６のチャンクが必要となる。マルウェア・エージェントは、この状態に関連付けられたＲＡＭの位置（それは必ずしもマルウェア・エージェントがこの状態を記憶した場所ではない）からハッシュ関数呼び出しのシーケンスを再計算しなければならない。

【００９０】
メモリ書き込みの際のセル上でのランダム置換（その順序をマルウェア・エージェントにより予測することはできない）を前提として、記憶された状態に対応するストリング位置までの距離の予想される長さは、少なくともｒｏｕｎｄｓ×ｎ／（ｃ／Ｌ）である。但し、ｎ＝ｎｕｍｂｅｒ＿ｂｌｏｃｋｓ×ｃｈｕｎｋｓ＿ｐｅｒ＿ｂｌｏｃｋはＲＡＭを成すチャンクの数に対応しており、ｒｏｕｎｄｓ×ｎは擬似ランダム・ストリングの長さであり、ｃ／Ｌはマルウェア・エージェントにより記憶される擬似ランダム状態の数である。このように、“不良”セルの各ヒットについて、マルウェア・エージェントは、ｎｅｘｔ＿ｓｔｒｉｎｇ＿ｃｈｕｎｋの予想されるｒｏｕｎｄｓ×ｎ×Ｌ／ｃの呼び出しを実行しなければならず、これはｒｏｕｎｄｓ×ｎ×／ｃに対応している。このようなヒットは、マルウェア・エージェントが予想される状態の１つを計算しようとする際に生じる“不良”セルのヒットはカウントすることなく、ｃ個ある。従って、マルウェア・エージェントは、記憶されている内容からｃ個の不良ブロックの内容を計算するため、少なくともｒｏｕｎｄｓ×ｎのハッシュ演算を実行しなければならない。（実装の一例に従って）これを行うためのおおよその時間は、正規のクライアントよりも、少なくとも１００，０００～１，０００，０００倍遅く、これは、欠落データを計算しようとする試みが検出されることを示している。

【００９１】
チェーンの計算によって、計算の別のパスのための値を記憶するのに使用されているセルへのアクセスが生じる場合、これによって別のヒットが生じる。それは、各メモリアクセスについて、およその確率（ｃ－ｃ／ｒｏｕｎｄｓ）／ｃ×ｃ／ｎｕｍｂｅｒ＿ｂｌｏｃｋｓ＝（ｃ－ｃ／ｎｕｍｂｅｒ＿ｂｌｏｃｋｓ）／ｎｕｍｂｅｒ＿ｂｌｏｃｋｓ≡ｃ／ｎｕｍｂｅｒ＿ｂｌｏｃｋｓで発生することになり、従って、上記のように、所与の最初の不良セル・ヒットについて、およその確率

【数１】

$$1-(1-c/number\_blocks)^{nubmer\_blockssrounds^{2}/c}$$

で発生することになる。この量の概算は、
【数２】

$$1-e^{-rounds^2}$$

である。ｒｏｕｎｄｓ＝２の場合、これは９８％超の確率である。この追加的コストは、ｃの値が増加すると増加することになる。このため、攻撃者は、ｃを小さくするように最善を尽くそうとする。
【００９２】
以下では、攻撃者がｃ＝Ｌ＝１６個のセルのみを使用し、１６個すべてを１つの値を記憶するために使用すると仮定する。この構成によると、“記憶セル”の値から“プログラム・セル”に対応する値を計算できるようにならない方向にチェーンがつながるような状況において、攻撃者は、（外部メモリを使用しない限り）値の計算に失敗することになる。ｒｏｕｎｄｓ＝２の場合、このような失敗が７５％の確率で発生する。残りの２５％の場合は、攻撃者は、単に速度が劣ることになる。攻撃者は、常に値の計算に成功するためには、それぞれ５１２ビットの長さである少なくともｒｏｕｎｄｓ＝２個の値を記憶する必要がある。
【００９３】
計算をアウトソーシングする－攻撃戦略３に対して防御する
【００９４】
一部の実施形態では、デバイス上の通信機器を用いて計算をアウトソーシングする時間がないように、チェックポイント間の時間が選択される。チェックポイント間の時間は、これを直ちに検出可能とするように、ベリファイア１０６により選択することができる。計算のアウトソーシングを伴う戦略は、ラウンドトリップが攻撃者により正しい値を提供するための２つのチェックポイントの間に完了しなければならないので、失敗する。これは、クライアント・デバイスと検証機関との間の通信速度には依存しない。
【００９５】
検出コードに変更を加える－攻撃戦略４に対して防御する
【００９６】
不正プログラム５０８は、（例えば、図６に関連して説明したように）一部のステップの実行に悪影響を及ぼすものであると仮定し、進んで正規のコードをロードして、自身を削除するものとする。このような攻撃者は、場合によってプロセスの部分６０２および６０４を破壊する可能性があるが、部分６０６を破壊することはできない。特に、アクティブであり続けるためには、プロセスの（スワップ領域およびＲＡＭをクリアする）部分６０２を破壊する必要がある。これによって、６０４で状態の誤ったレポートを引き起こすことが可能となる。しかしながら、これは、メモリ内容の鍵付きハッシュが計算されたとき（６１０）に検出される。これは、使用されるハッシュ関数の衝突がないことが仮定されていることと、６０８までは鍵がデバイスに開示されないことと、この両方に起因する。部分６０８は、６０６の間にアクティブでなければ破壊されることはあり得ず、これによって、上記のように検出が生じることになる。また、回避的プログラムは、図６に示すプロセスの部分６０６を実行することなく、６１２で報告される正しい値を計算することはできない。
【００９７】
４つの攻撃戦略の組み合わせでも、それらの各々が検出され、それらの組み合わせにより基本的なデバイス固有の制限は変更されないので、失敗する。
【００９８】
追加処理
【００９９】
図８は、デバイスの監査を実行するプロセスの例を示している。様々な実施形態におい

て、上記の監査プロセスは、２（またはそれ以上の）段階のプロセスの一段階（８０２）を成している。デバイス１０２などのデバイスに関して上記の技術が採用されていれば、デバイスのＲＡＭ内にアクティブな回避的ソフトウェアはないと仮定することができる。そこで、任意の追加処理をデバイス上で実行することが可能である（８０４）。実行することができる追加処理の例について以下で説明する。

【０１００】

例：マルウェア

【０１０１】

８０２の処理を実行した後に、デバイス１０２は、８０４において従来のアンチウイルス・ソフトウェアを実行し、これにより、メモリ２０６に記憶されている可能性のあるものなど、既知の不正ソフトウェアを識別する。８０４において、デバイス１０２は、さらに、メモリ２０６の内容全体またはメモリの部分について、ベリファイア１０６または他のデバイスにレポートするように構成することもできる。

【０１０２】

例：ジェイルブレイク

【０１０３】

８０２の処理を実行した後に、デバイス１０２は、８０４において、そのオペレーティング・システム・ローダが特定のハッシュを持つかどうか判断し、および／または、そうでなければオペレーティング・システム・ローダが望ましい状態から変更されているかどうか判断する。

【０１０４】

例：電話機のロック解除

【０１０５】

８０２の処理を実行した後に、デバイス１０２は、８０４において、そのオペレーティング・システム・ローダが変更されているかどうか判断し、サービス・プロバイダに関連付けられた情報が変更されているかどうか判断する。

【０１０６】

例：ソフトウェアの不正コピー

【０１０７】

８０２の処理を実行した後に、デバイス１０２は、８０４において、メモリ２０６内に含まれるソフトウェアが期待される設定から変更されているかどうか判断し、そのソフトウェアに関連するシリアル番号を確認し、および／または、そうでなければ、含まれるソフトウェアが不正に／ライセンスを受けずに用いられているかどうか判断する。一部の実施形態において、デバイス１０２は、メモリ２０６の内容またはその一部についてベリファイア１０６にレポートする。

【０１０８】

例：メディアの不正コピー

【０１０９】

メディア・ファイル（例えば、音楽ファイル、動画ファイル、または画像ファイル）は、配信時に透かしを使用してカスタマイズされると仮定し、これらの透かしは、例えばＭＡＣまたはデジタル署名を使用して暗号的に認証されるものとする。８０４において、デバイス１０２上にあるどのファイルが正規の透かしを持っているか、また、これらは有効な認証コードを含んでいるかどうか、判断することができる。この判断は、デバイス１０２でローカルに行うか、あるいは中央で（例えば、ベリファイア１０６において）行うか、いずれかが可能である。

【０１１０】

様々な実施形態において、（デバイス１０２にインストールされている音楽プレーヤなどの）アプリケーションが、（アクティビティのログを形成して）使用状況およびその他のデータを記録し、その情報を適当なメディア（例えば、曲ファイル）と関連付ける。ログは、８０４においてベリファイア１０６により読み出すことができる。

【０１１１】
例：管理の連鎖／使用状況ログ
【０１１２】
あるアプリケーション（またはデータ・ファイル）は、これに関連付けられて、トランザクションを記録するのに使用されるログ・ファイルを持つと仮定する。一つの例は、ストアド・バリュー情報を含む金融取引の発生を記録するログ・ファイルである。ログ・ファイルに加えられた変更の正当性は、次のように検証することができる。まず、８０２の処理が実行される。次に、８０４において、そのアプリケーション（またはデータ・ファイル）が変更されているか否か、ひいてはログ・ファイルが本物であるかどうかについて、（例えば、プログラム・イメージのハッシュを比較することにより）判断することができる。
【０１１３】
この例において８０４で実行される処理の一つのアプローチは次のとおりである。まず、メモリ２０６がスキャンされ、アプリケーションおよびアプリケーションに関連付けられたデータ・ファイルのリストが作成される。次に、アプリケーションおよびデータ・ファイルの記述子のリストが決定される。記述子の例は、ファイルの名前およびタイプを加えたファイルのハッシュ、および、それがどのストリング・シーケンス（複数の場合もある）とマッチしたかを示す識別子である。次に、第１のリストの中にまだレポートされていないアプリケーションまたはデータの記述からなる第２のリストが作成される。ここで作成される記述には、アプリケーションのコード、または、それが処理する入力ファイルおよび生成する出力ファイルの種類についての記述の、全てまたは一部が含まれることがある。第２のリストは、ベリファイア１０６などの外部機関に送信されて、そこで検証される。また、第２のリストは、ポリシー検証サーバから取得したポリシーを用いてローカルに処理することもできる。
【０１１４】
検証の結果は、アプリケーションおよびデータへのアクセス許可に影響を及ぼすように用いることができ、さらに、（インターネット、３Ｇネットワーク、企業ネットワークなどの）ネットワーク資源へのアクセスが認められるかどうかを含めて、外部サーバがデバイスとどのように情報のやりとりを行うかを制御するために用いることができる。別の例として、デバイス上での実行が許可されるソフトウェアを制限することができ、そして、コンプライアンスの欠如をユーザに通知し、ファイルを削除もしくは修復、またはそうでなければ変更することを試みるなどすることができる。
【０１１５】
例：ペアレンタル・コントロール・フィルタおよびその他のモニタリング機能
【０１１６】
８０２の処理が実行された後に、様々な実施形態において、追加のミドルウェアがインストールされ、これは、デバイスに関連する様々な事象をログ記録（および／または阻止）するように構成することができる。例として以下のものが含まれる。
【０１１７】
（ａ）デバイスで生成され、後に送信された写真を、（例えば、“セクスティング”を防ぐために）確認する。
【０１１８】
（ｂ）時速２０マイルを超える速度で走行しながらデバイスが（例えば、テキスティングまたはビデオクリップを見るために）使用されたかどうか、（例えば、デバイス・アクティビティおよびＧＰＳの変化に基づいて）判断する。
【０１１９】
（ｃ）（デフォルトのプログラムの他に第２のインスタント・メッセージ・プログラムなど）代替アプリケーションがインストールされているかどうか、（例えば、インストール・アクティビティに基づいて）判断し、そして代替アプリケーションについてのログ・ファイルを作成する。

【０１２０】
（ｄ）ユーザがどのようなＵＲＬを訪れたのかについて、どのＵＲＬには手動ナビゲートにより到達し、どのＵＲＬにはアクセスされた他のＨＴＭＬ文書内のリンクにより到達したかを含めて、（例えば、ブラウザの履歴情報に基づいて）判断する。このロギングの一つの利点は、人がフィッシングの被害に遭った可能性があるかどうか、マルウェアを含めて望ましくないコンテンツを配信することが知られているウェブサイトを訪れたかどうか、さらには、デバイスがクリック詐欺に巻き込まれた可能性があるかどうか、特定されることである。このような悪用は、例えば、ＪａｖａＳｃｒｉｐｔ（登録商標）、カスケーディング・スタイルシート、および／または他の関連するスクリプト言語を使用することにより、デバイス自体を感染させることなく達成することが可能である。
【０１２１】
例：追加の適用例
【０１２２】
本明細書に記載の技術は、上記の例に加えて、さらに多くの利用が可能である。例えば、デバイスの監査は、使用状況、保険の目的、関税、税金、通行料などを測定するための車両用ブラックボックスにおいて、マルウェアと意図的な不正使用の両方を特定する目的で用いることができる。
【０１２３】
デバイスを監査する技術は、他のアプリケーションのコンポーネントとして組み込むことができ、これらのアプリケーションは、スキャンを実行するために自身を一時的にサスペンドすることが可能であり、後に、感染のないことがわかっている状態において再び制御を渡される。
【０１２４】
さらに別の例として、この技術を、医療機器において、それらが感染しておらず、正しく設定され維持されていることを確認するために用いることができ、さらに、誰がデータおよび機器にアクセスしたのかを知ることが重要となる特別な場合において使用状況を監査するために用いることができる。当該機器は、使用状況の情報を常にログ記録する場合があり、これは、プリロード・アプリケーションが干渉しないように行われる。監査プロセスは、プリロード・アプリケーションが良好な状態のままであること、および競合するアプリケーションまたは設定が存在しないことをアサートするためのメモリ印刷スキャンを含んでいる。
【０１２５】
最後に、修正の必要がない、あるいはそれが主な目的ではない状況におけるマルウェアの検出に、本技術を用いることができる。そのような状況の一つは、オンライン・ゲームにおいて、ゲームで不正をするためのモジュールが無いことを検出する場合である。
【０１２６】
プライバシーの保護
【０１２７】
一部の実施形態では、全ての状態の記述（例えば、メモリ２０４の内容）がベリファイア１０６に伝えられる。しかしながら、秘密鍵および非実行可能プライベート・データなど一部のデータは、好ましくは、デバイス１０２から転送されるべきではない。次のセクションでは、そのようなデータのプライバシーを保護する技術について説明する。
【０１２８】
第１の乱数をｘと呼ぶものとし、これは、ある可能な値１．．．max_xの空間から選択されると仮定する。ｘは、その目的であった監査プロセスへの入力を提供するだけではなく、マルウェアを符号化することが可能である。正規のプログラムは、入力データｘ、および（g_1，n_1）と呼ばれるいくつかのシステム・パラメータから、一方向関数の値ｙを計算する。これを行う方法の一つの例は、$y = g_1^x \bmod n_1$を計算することによるものであり、ここで、g_1は大きなサブグループG_{n1}を生成する。
【０１２９】

次に、プログラムにより、値ｙ、および（g_2，n_2）と呼ばれるいくつかのシステム・パラメータから、第２の一方向関数の値ｚを計算する。これを行う方法の一つの例は、ｚ＝$g_2^y \bmod n_2$を計算することによるものであり、ここで、g_2は大きなサブグループG_{n2}を生成する。

【０１３０】

次に、（ｚ，g_1，g_2，n_1，n_2）がベリファイアにより知られており、（ｚ，ｘ）が知られていないとして、クライアント・マシンは、

【数３】

$$z = g_2^{g_1^x \text{ modulo } n_1}$$

となる値ｘがあることを、（例えば、ゼロ知識証明を用いて）証明すると仮定する。そして、デバイス（“証明者”）は、値ｘを消去するが、（ｙ，ｚ）およびパラメータ（g_1，g_2，n_1，n_2）は保存する。

【０１３１】

その後に、デバイスは、自身が保存しているが秘密である値ｙが、値ｚに対応していることを証明しなければならない。（ここで、ｚはデバイス１０２上に保存することができるが、ベリファイア１０６により保存することも可能である。）用いることができる証明の一例は、ゼロ知識証明である。

【０１３２】

第２の証明が終わり、ベリファイア１０６がそれを受け入れると、クライアントが記憶している、知られていない値ｚが、マルウェア・エージェントに対して多くの量の値のデータを隠蔽するのに用いることができない形式であることを、ベリファイアが知る。

【０１３３】

ここで、ｚは、ｍと呼ばれる他のデータを暗号化するために用いることができ、その暗号文はｃと呼ばれる。従って、暗号化アルゴリズムＥについて、ｃ＝E_z（ｍ）である。対称暗号化と仮定すると、復号アルゴリズムＤについて、ｍ＝D_z（ｃ）である。デバイスの内容の検証は可能であるが、ｍは、ｃを受け取った機関により知られていないままである。この機関は、ｚを知らないが、ｚが大量のマルウェア・データを隠蔽することができない何らかの受け入れ可能な形式のものであることのみ知っている。本明細書に記載の監査プロセスによって、検証機関は、クライアント・デバイスのＲＡＭ内に正規のプログラムのみが存在すると確信することができるので、プログラムは、ｃが与えられると、―秘密の値ｚを用いて―ｍにアクセスすることが可能であることが分かる。しかし、ベリファイアは不可能である。

【０１３４】

アクセスするプログラムが正規のものであることが分かっているので、ｍは、認められた方法でのみアクセスされることも分かっている。例えば、ｍがデータであってコードではない場合、アクセスするプログラムは、データを実行しようとしない。

【０１３５】

擬似ランダム・ストリング発生器の利用

【０１３６】

図９は、デバイスの監査が行われる環境の一実施形態を示している。図示の例では、デバイス９０２は、図２に示す構成要素に加えて、外部のベリファイア９０４のプロキシ（９０６）として機能するように構成されたＳＩＭを備えている。以下でより詳細に説明するように、デバイス１０２の命令キャッシュに記憶されている（そこにその全体が収まっている）モノリシック・カーネルは、起動されると、（除外するものとして選択したプロセスを除いて）他のすべてのプロセスをスワップアウトして、監査プロセスを実行する。モノリシック・カーネルは、データ・キャッシュ（およびレジスタ）に配置された関連の作業領域を有している。キャッシュは、一般的にＲＡＭを用いて実装されており、ここで

はその一部と見なされる。本明細書において用いられる“空きＲＡＭ”とは、通常のカーネルを含む全てのアプリケーションがスワップアウトされた後に空きになるはずのＲＡＭ部分である。一部の実施形態では、“空きＲＡＭ”は、承認されたルーチンおよびデータのセットによって占められていないＲＡＭのセグメントとして定義される。例えば、通常のカーネルは、承認されたルーチンとなる場合があり、共通アプリケーションおよびホワイトリストにあるアプリケーションもそうである場合がある。また、承認されたデータは、外部ベリファイアにより知られているデータに対応していることがあり、それはホワイトリストにある（つまり、安全と考えられる）のであれば、どのような形式であってもよい。これらの場合、承認されたプログラムは、（以下でより詳細に説明するように）二次記憶装置にスワップアウトされる必要はなく、その代わりに、監査のメモリ読み取り部分（例えば、１１０８）の間、常駐したままでいることができる。

【０１３７】

一部の実施形態では、モノリシック・カーネルは、既知の実行環境εに対してパラメータ化されたプログラムF_εに相当する。上記で説明したように、実行環境は、デバイスのハードウェア構成に対応している。入力ｘについてF_εを実行すると、出力のシーケンス$F_{\varepsilon i}(F_\varepsilon, x)$が、それぞれ実行開始から時間$t_i(F_\varepsilon, x)$内に生成され、そして終了状態$s(F_\varepsilon, x)$が生成される。この例では、Ｘが全ての正規の入力のセットであるとして、$x \in X$である。

【０１３８】

プロキシ９０６は、デバイスからの待ち時間の変動を低減するために用いられるものであり、様々な実施形態において、ＳＩＭに代えて、またはこれに加えて、テザリングされた携帯電話機、携帯電話の中継塔などとして実装される。一部の実施形態において、外部のベリファイア９０４は、（以下でより詳細に説明する）初期計算を実行し、仲介役としてデバイス９０２を用いて情報の一部をプロキシ９０６に（例えば、セキュア・チャネルを介して）伝える。プロキシ９０６は、モノリシック・カーネルにより実行される計算の時間を計測し、時間計測値についてのレポートを外部のベリファイア９０４に返す。一部の実施形態では、プロキシ９０６の代わりに、またはこれに加えて、テザリングされた携帯電話機またはコンピュータといった外部デバイス、基地局、もしくは追加の外部ベリファイアが用いられる。また、不正に強いと考えられているソフトウェア・プロキシを用いること、または専用ハードウェア・プロキシを用いることも可能である。

【０１３９】

図１０は、デバイスの一部分の一実施形態を示している。前述のように、“空きＲＡＭ”は、全てのアプリケーションおよび標準カーネルがスワップアウトされた後に空きになるはずのＲＡＭ部分であると定義される。バスの幅は１ワードである。メモリの大きさも、やはりワードを単位として記述可能である。例えば、標準的な電話機では、１ワードが３２ビットで、図１０に示すような５１２バイトのメモリページは１２８ワードの大きさである。ここで用いる“チャンク”とは、キャッシュラインの長さである。図示の例では、各ワードが３２ビットで、キャッシュラインは８ワードに相当し、従ってチャンクは２５６ビットである。

【０１４０】

図１１は、デバイスの監査を実行するプロセスの一実施形態を示している。様々な実施形態において、図１１に示すプロセスは、デバイス９０２により実行される。プロセスは、その計算が特定の時間で完了することが期待されるように構成される。評価される空きＲＡＭの容量に変化があること、および二次記憶装置１００４にアクセスしようとする試行があることによって、計算が完了するまでにかかる時間に観測可能な延長が生じる。同様に、ホワイトリストにあるプログラムのいずれか、または関連データの内容が変更されると、遅延または誤った応答が計算される原因となる。

【０１４１】

図１１に示すプロセスは、図３に示すプロセスに関連して説明した方法など様々な方法で、起動することができる。追加の例として、監査プロセスを、シャットダウンまたはブ

ート・ルートに含めることができる。また、アプリケーションにより監査プロセスを起動させることも可能である。アプリケーションが非アクティブにされ、処理が実行されて、完了すると制御がアプリケーションに戻される。一部の実施形態において、アプリケーションは、最後のスキャンがどれほど最近に実行されたのかについての情報を、中央権限（または当該デバイス）に問い合わせる。ＳＩＭカードは、スキャンがいつ実行されたのかについての情報を記憶することができる。ＳＩＭカードが、時間の常時測定を可能にする機能を持つ場合は、回答として実際の時刻を与えることができる。それ以外の場合は、その多くは定期的であることが知られている、認識されたトランザクションの数に基づき、推定の時刻を与えることができる。このような情報を、最後のスキャンからの時間を評価するために用いることができる。

【０１４２】

プロセスは、モノリシック・カーネル１００６（および、保持することが容認できると見なされるプロセス）を除いてメモリ１００２の内容が二次記憶装置１００４にスワップされる１１０２で開始する。一部の実施形態では、プロセスの部分１１０２は、通常のカーネルまたはその部分のスワップアウトを含んでいる。シリアル通信用のデバイスドライバなど不可欠の機能は、モノリシック・カーネル１１０６に再び組み込まれる。様々な実施形態において、その内容はそのまま逐語的にスワップアウトされるか、または内容のコンパクト記述が、スワップアウトされるか、プロキシ、外部のベリファイア、もしくは他の信頼済みのデバイスに記憶されるか、もしくはアクティブなコードに使用することができない状態でＲＡＭに記憶される。（例えば、命令用ではなくデータのためのみのキャッシュ部分に命令を記憶することが可能である。）一部の実施形態では、“空き”領域が存在せず、図１１に示すプロセスの部分１１０２は省略される。

【０１４３】

１１０４において、ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取る。プロセスのこの部分は、図３に示すプロセスの部分３０２と同様である。１１０４では、さらに、擬似ランダム・ストリングを生成するために使用することができるシードなど、初期化情報を受け取る。初期化情報の他の例には、以下でより詳細に説明するように、ステップ値およびキー値が含まれる。

【０１４４】

１１０６において、空きＲＡＭに上書きする。一部の実施形態では、シードを用いて生成される擬似ランダム・ストリングの出力を使用して、空きＲＡＭに上書きする。空きＲＡＭに上書きする１つの方法は、n^2ビットの擬似ランダム・ストリングであって、その出力ビットはいずれの１つもその計算に、ビットのブロック全体の計算または５１２ビット・モードでのＭＤ６の少なくとも５１２の適用の少なくとも半分の時間がかかるという特別な特性をもつものを生成することである。この方法は３段階を使用し、出力ストリングが空きＲＡＭの全体を埋めるまで（異なるａｕｘ値で）繰り返される。

【０１４５】

１．生成：“ｎ”ビットの出力サイズのハッシュ関数ｈ（例えば、ＭＤ６）を用いて、$0 \leqq i \leqq n-1$、および、ある値ａｕｘについて、$x_i = h(seed, i, aux)$を生成する。これは、n^2の擬似ランダム・ビットを生成する。

【０１４６】

２．シャフリング：$y_j = \Pi_{i=0} 2^i BIT_j(x_i)$，$0 \leqq j \leqq n-1$，を計算する。ただし、$BIT_j$は、入力のｊ番目の最上位ビットを返す関数である。これによって、値のいずれの１つを再構成するためにも全てのｎのハッシュ関数適用を計算することが必要となるように、ビットがシャッフルされる。

【０１４７】

３．ブレンディング：$0 \leqq j \leqq n-1$について、$z_j = h(y_j)$を計算する。これによって、出力の各ビットが、全てのｎ個の入力ビットの関数であることが保証され、その各々は計算に１つのハッシュ関数評価を必要としたものである。

【０１４８】

様々な実施形態において、最終的なストリングの任意の部分を計算するコストをさらに増加させるため、追加のシャフリングおよびブレンディングが実行される。さらに、図１１に示すプロセスの部分１１０６に関連して説明した技術の例の代わりに、空きＲＡＭを上書きする他の技術を用いることができる。

【０１４９】

１１０８において、“ステップ”値により決定される方法でメモリ１００２（またはその一部）が読み取られる。結果は累積され、計算結果にはキーを用いて鍵がかけられる。様々な実施形態において、部分１１０８の処理は、それぞれ以下でより詳細に説明されるメモリアクセス・スケジューラとアキュムレータにより実行される。

【０１５０】

メモリアクセス・スケジューラ

【０１５１】

“ｓＲＡＭ”を、チャンクを単位として測定されたＲＡＭ １００２全体のサイズとする。外部ベリファイア９０４は、ｐａｇｅ＜ｓｔｅｐ＜ｓＲＡＭ－ｐａｇｅの範囲で、乱数値ｓｔｅｐを、“ｓｔｅｐ”が奇数値であるように選択する。ここで、“ｐａｇｅ”は二次記憶装置の１つのメモリページのサイズを示しており、同じくチャンクを単位として測定されたものである。複数のページサイズがある場合（例えば、二次記憶装置を構成するコンポーネントが複数である場合）、様々な実施形態において、最大のページサイズが用いられる。

【０１５２】

１１０８の処理の実行は、メモリにアクセスするループを含み、その結果を結合して鍵付きのメモリチェックサムを形成することを含む。ループの繰り返しごとに、アクセス位置は、ｓＲＡＭを法として値ｓｔｅｐずつ増加される。“ｓｔｅｐ”とｓＲＡＭは互いに素であるので、ＲＡＭの全てのメモリ位置がちょうど１回ずつアクセスされる。また、アクセス順序は、値“ｓｔｅｐ”が開示されるまで、攻撃者に知られることはない。“ｓｔｅｐ”に従って読み取られるメモリ１００２の図解を図１２に提示している。

【０１５３】

図９に示す例では、デバイス９０２は、１つのシングルコアＣＰＵを有している。マルチコア・プロセッサおよび／またはマルチプロセッサを有するラップトップ・コンピュータなどのシステムでは、１１０８の処理は、本質的にシリアルとなるように（従って、これによりマルチプロセッサの利用を阻むように）構成するか、またはマルチプロセッサを活用するように構成するか、いずれかにすることができる。後者の一例として、衝突がなく、各スレッドがメモリの異なる部分に対応するように、いくつかの計算を、オフセットを伴って開始させることができる。

【０１５４】

アキュムレータ

【０１５５】

前のレジスタ内容（ここでは“状態”と呼ぶ）を、新たに読み取ったメモリ内容（データ）と結合させる単純な非線形関数を用いて、メモリの内容をレジスタに１つずつ累積することができる。累積関数の例として、ハッシュ関数（例えば、ＭＤ６）、非線形シフトバック・レジスタ、およびより単純な関数が含まれる。

【０１５６】

単純な関数の例は、ｓｔａｔｅ←ＲＯＲ（ｓｔａｔｅ＋ｄａｔａ）である。後者の関数は、関数ＲＯＲ（．．．（ＲＯＲ（$state_0$＋$data_1$）＋$data_2$）．．．＋$data_n$）に対応しており、ここで、“＋”は通常の加算を指し、“ＲＯＲ”はレジスタの内容を１ビット右に循環させる。この場合、関数自体は非線形ではないかもしれないが、アプリオリには未知であるｓｔｅｐサイズ、および厳しいタイミング要件と組み合わせると、それでも必要な処理要件を満たすのに十分である。

【０１５７】

前述のように、様々な実施形態において、累積プロセスは鍵がかけられる。これを達成

する一つの方法は、一定の間隔で、値“ｓｔａｔｅ”を、（外部ベリファイアまたはプロキシから取得される）新しい“ｋｅｙ”値でオフセットさせることである。オフセットは、新しいｋｅｙ値に現在の値ｓｔａｔｅを加算することにより実行することができる。

【０１５８】

また、１１０８に関連して説明したプロセスはメモリの読み取りに基づいているが、一部の実施形態では、フラッシュメモリによるさらなる速度低下を生じさせるように、書き込み操作が含まれる。一例として、データがフラッシュメモリに記憶されているのであれば、ブロック全体を消去するものとして、“１”のシーケンスが書き込まれる。書き込む場所のスケジューリング（および、その場合のモノリシック・カーネル）を簡単化するため、場所は、新しいｋｅｙ値を取得すると同時にプロキシから取得することができる。

【０１５９】

１１０８では、さらに、他の様々なメモリアクセス・シーケンスを実行することができる。例えば、１つではなく２つのｓｔｅｐ値を用いることが可能であり、これらのｓｔｅｐ値は両方とも偶数とすることができるが、これらはほとんどの場合、空間全体をカバーするものである。場所のシーケンスを選択する関数を決定する数またはパラメータの集まりを使用することも可能である。これは、最大長シーケンスと考えることができ、その出力は場所であって、最大長シーケンスには所定の範囲内のすべての値が含まれ、それらはメモリ位置に対応している。もし、特定の領域（例えば、モノリシック・カーネル）へのアクセスを避けることが望ましいのであれば、そうなるように、それらの値をオフセットさせることができる。最大長シーケンスの場合、外部ベリファイアまたはプロキシにより提供されるキーは、初期状態であるか、あるいはＬＦＳＲの様々なセルに関連付けられた重みとすることができる。

【０１６０】

１１１０において、鍵付きの計算結果が外部ベリファイア９０４に提供される。外部ベリファイアが結果を承認した場合、デバイス９０２は安全な状態にあるとみなされる。

【０１６１】

１１１２において、デバイス９０２は、安全な状態で実行されるべきあらゆる機能を実行する。例として、ＳＳＬ接続の設定、投票、パスワードの入力、二次記憶装置の悪質な／望ましくないプログラムのスキャンなどが含まれる。様々な実施形態において、安全状態の機能のコードが二次記憶装置内にある（すなわち、モノリシック・カーネルの一部ではない）場合、その機能のダイジェストが、モノリシック・カーネル（またはプロキシ）に記憶されている値と比較される。それらの値が一致している場合のみ、機能が有効にされる。様々な実施形態において、プロキシがメッセージ・ダイジェストの計算を実行可能である場合、モノリシック・カーネルは同様のことを行うためのコードを含む必要はない。

【０１６２】

１１１４において、（１１０２でスワップアウトした）二次記憶装置１００４の内容をロードすることにより、ＲＡＭ １００２の状態が復元される。

【０１６３】

図１１に示すプロセスにおいて可能性のある負荷の大部分は、ＲＡＭから二次記憶装置へのアプリケーションおよびデータのスワップアウトと、そのスワップインにかかわるものである。これを行うことを、例えば時間を節約するために回避することが可能である。これは、アプリケーションをキルすることにより実現することができる。どのようなアプリケーションが実行中であるか、さらに場合によってはそれらの状態またはその一部について、外部ベリファイアまたはその他の資源が知っている場合、この機関は、監査プロセスが実行された後に、選択されたアプリケーションの再起動を支援することが可能である。二次記憶装置またはＳＩＭカードまたは他のオンボード・ユニットは、この情報の一部を保持することが可能である。空間と時間を節約するため、アプリケーションおよびデータは、それらの完全なストリングによってではなく短縮された識別子によって識別することが可能である。検出アルゴリズムが実行された後に、同じ状態をほぼ再生する近似アル

ゴリズムをもつことが可能である。例えば、これはブラウザを再起動するが、ブラウザの内容を復元できない場合がある。

【０１６４】

また、アクティブなアプリケーションは、それらがＲＡＭのある部分のみを占めるのであれば、スワップアウトする必要はない。例えば、それらがＲＡＭの前半のみを占めると仮定する。空きＲＡＭ内の各セル（番号ｉ）について、そのセルの内容をより後ろの位置（位置２ｉ）にコピーする。これは、好ましくは終わり（大きい番号の位置）から開始して実行される。これにより、アプリケーションは効果的にスライスされて、偶数位置にのみに存在することになる。この場合、擬似ランダム値は、奇数番号の位置にのみ書き込まれる必要があり、また、奇数番号のセルについてのみ非線形累積を実行する必要がある。注目されるのは、いずれの機能的マルウェアもアクティブのままでいることが不可能であるということである。マルウェアが存在することは可能であるが、それは、その命令が“次の空き空間へのジャンプ”であって、そこが次の命令がある場所である場合に限られる。擬似ランダムなものにより上書きされない全ての空間はジャンプのみとなる（それ以外のための空間は連続空間にない）ので、マルウェアは何も達成できないことがわかる。特に、空間が攻撃者により予測可能でない場合は、スライス間の距離をより大きくすることが可能である。距離は、例えば、シーケンス発生器により予測される場合があり、このとき、異なる距離は異なる長さである。ＲＡＭ内でのデータおよびプログラムの展開は、これらをランダム・ストリングでオフセットさせることと組み合わせることができる。マイクロカーネル（マルウェア検出を担うプログラム）は、実行が可能な状態のままである必要があるので、このように展開されることはない。

【０１６５】

図１１に関連して、アプリオリに知られていないｓｔｅｐサイズを用いて、いかにしてＲＡＭを読み取ることができるかについて説明を行った。１以外のｓｔｅｐサイズを用いて（空き）ＲＡＭに書き込みをすることも可能であり、この場合のｓｔｅｐサイズはアプリオリに知られていないものであってよい。この効果の一つは、これによって、攻撃者が二次記憶装置に情報を保存したいと望む場合に、二次記憶装置への書き込みが生じることである。この場合の遅延は、二次記憶がフラッシュメモリを使用する場合、読み取り遅延よりも大きい。書き込みされる範囲に対応する整数を法とする単純なｓｔｅｐインクリメントを用いることが可能であり、モノリシック・カーネルが上位アドレスに常駐していない場合は、さらにオフセットが用いられる。演算に特定のモジュロを用いないこと―アドレス可能ＲＡＭ空間に対応するモジュロを使用することに相当する―も可能であり、書き込みされるセルが、書き込みされるべき範囲にあるかどうか識別される。

【０１６６】

図１３は、選択的にメモリを読み取るプロセスの実装の一実施形態を示している。一部の実施形態において、図１３は、図１１に示すプロセスの部分１１０８の一部を実現するために用いられる。

【０１６７】

一部の実施形態では、データ・キャッシュがフルに利用される。具体的には、データは、キャッシュライン幅であるチャンクで処理され、そのサイズは通常８から５１２バイトに及ぶ。以下では、パラメータ“α”はキャッシュライン幅をワードで示している。パラメータ“ｐｅｒｉｏｄ”は、計時周期の長さに相当する。これは、デバイス９０２から可能性のある外部ヘルパーへの楽観的なラウンドトリップ待ち時間よりも短い時間に対応するように設定される。報告された値と、それらが報告された時刻が、プロキシ９０６により記録される。

【０１６８】

様々な回避的プログラムの検出例

【０１６９】

以下のセクションでは、本明細書に記載の技術を用いて、上記の様々な戦略を用いる回避的プログラムをいかにして検出することができるかについて、例を提示する。検出をう

まく回避するためには、攻撃者は期待時間$t_i(F_\varepsilon, x)+\delta$内に正しい応答$F_{\varepsilon i}(x)$を提供しなければならない。これを攻撃者が試みることができる様々な方法があり、その各々について以下で説明する。

【０１７０】

外部支援－攻撃戦略１に対して防御する

【０１７１】

値“ｋｅｙ”および“ｓｔｅｐ”が、セキュア・チャネルを介してプロキシに伝えられる。“ｓｔａｔｅ”の値が、外側ループの繰り返しごとに報告される。この値の計算は、値“ｋｅｙ”に依存する。従って、（値ｋｅｙを送信してから計算結果を受け取るまでの）ラウンドトリップ待ち時間が時間閾値$t_i(F_\varepsilon, x)+\delta$を超える場合、外部デバイスは、応答の生成を支援することが不可能である。ここで、$t_i(F_\varepsilon, x)$は、F_εが応答を生成する時間である。

【０１７２】

二次記憶装置を用いる－攻撃戦略２に対して防御する

【０１７３】

攻撃者がＲＡＭ内のデータの“ｋ”個のワードに変更を加えたとし、この領域の“意図的な”内容のイメージが二次記憶装置に保持されていると仮定する。変更されたセルの１つにF_εがアクセスしようとした場合、代わりにＦ’が二次記憶装置内のイメージにアクセスする。

【０１７４】

これは、３つの方法で行うことができる。（ａ）ＲＡＭ全体のイメージを二次記憶装置内につくることができ、各アクセスはそこで行われる。（ｂ）F_εの各メモリアクセスは、そのアドレスを範囲と比較することにより、単に二次記憶装置に転送する選択アクセス試行で置き換えられる。（ｃ）アドレス変換バッファ（ＴＬＢ：Ｔｒａｎｓｌａｔｉｏｎ Ｌｏｏｋａｓｉｄｅ Ｂｕｆｆｅｒ）を用いて、アクセスを自動的に選択された範囲に転送する。

【０１７５】

第１のケースでは、各メモリアクセスは、ＲＡＭの代わりに二次記憶装置にアクセスすることに伴う遅延が生じる。第２のケースでは、この遅延が転送されるアクセスで生じるだけでなく、各メモリアクセスについて比較を行う必要がある。第３のケースでは、転送されるアクセスについてのみ遅延が生じる。しかしながら、ＴＬＢの粒度がやや粗いことによって、アクセスの転送が可能なセルは最少数である。

【０１７６】

ＲＡＭを用いる－攻撃戦略３に対して防御する

【０１７７】

各セルの値は、（一部の実施形態では）第２層のハッシュ関数出力の一部であり、その入力は５１２の第１層ハッシュ関数評価の関数であるため、空きＲＡＭを埋めるのに用いられる擬似ランダム・ストリングの構築によって、いずれの１つのセルの計算も、少なくとも５１２のハッシュ関数評価の計算を必要とする。

【０１７８】

各ハッシュ関数の計算のセットアップ・コストがあって、それが少なくとも１１５６クロック・サイクルであると仮定する。同時に、ＭＤ６‐５１２のハッシュ計算は－セットアップが完了していれば－、１バイトあたり１５５サイクル、または６４ビット入力について９９２０サイクルかかる。これらは両方とも、最適化されたコードと、標準的な３２ビット・プロセッサの場合を想定している。従って、この設定では、１つのハッシュ関数評価あたりの総コストは１１０７６サイクルである。ＲＡＭ内の１つのセルのみの内容の計算で、少なくとも（５１２＋１）＊１１０７６サイクルかかることになる。

【０１７９】

時間計測へのプロキシの利用

【０１８０】

前述のように、一部の実施形態では、新しいシードを配布し、計時チェックポイントからの値の着信時間を計測するために、外部ベリファイアが用いられる。監査されるデバイスと外部ベリファイアとの間の通信モデルの２つの例は、（ａ）物理的接続、および（ｂ）外部ベリファイアとＳＩＭカードとの間のＶＰＮ、である。下りのデータは暗号化され、上りのデータは認証され、および／または双方向認証が行われる。

【０１８１】

デバイスとベリファイアとの間で物理的接続が用いられない場合、待ち時間の変動が、計算の時間計測の妨げとなり得る。このため、一部の実施形態では、（待ち時間の変動がより少ない）プロキシを用いて時間計測を支援する。プロキシとして用いることができるハードウェアの一例は、ＳＩＭカードである。

【０１８２】

計算の開始に伴う変動を低減するためにプロキシを用いることができる。これは、計算タスク全体とサブタスクの両方に当てはまる。これらは、両方とも、計算に必要なキーまたはシードを送信することにより開始させることができ、キーまたはシードを受信するまでは計算が開始されないことが保証される。

【０１８３】

ＳＩＭカードは暗号化されたシードを受け取って、それらを復号し、デバイス９０２に値を提供する。この配信時間は、デバイス上で観測される他の事象に関連させることができる。トリプレット（ｌｏｃａｔｉｏｎ，ｄａｔａ，ｓｅｅｄ）を考える。ベリファイアは、このようなトリプレットを生成し、それらを暗号化と認証による方法でＳＩＭカードに送信すると仮定する。ここで、ｌｏｃａｔｉｏｎは、プログラム内での計算ステップ（または段階）を記述し、ｄａｔａは、このｌｏｃａｔｉｏｎに関連付けられた何らかの状態を記述している。計算は外部ベリファイアにより先読みすることができる、または非常に可能性の高い少数の計算パスを先読みすることができると仮定する。

【０１８４】

これによって、外部ベリファイアは、所与のｌｏｃａｔｉｏｎ（すなわち段階）で、どのようなｄａｔａ（すなわち状態）がデバイス上で観測可能となるのかを予測して、このようなトリプレットを計算することが可能である。トリプルの第３の要素であるｓｅｅｄは、ｌｏｃａｔｉｏｎおよびｄａｔａと関連付けられた特定の状態に達した場合のデバイスに配信される値を示している。

【０１８５】

ＳＩＭカードは、値“ｌｏｃａｔｉｏｎ”をデバイスにより計算または予測することができないか、または他の所から受け取ることができないのであれば、これをデバイスに送る。計算が値ｌｏｃａｔｉｏｎ（これはループ反復値とすることができる）に関連付けられた段階に達すると、デバイス上のソフトウェアは、最も最近に計算されたある所定のタイプの値―またはその一部―を、ＳＩＭカードに送る。これは、最も最近に計算された値、所定のレジスタの内容、または他の適当な値とすることができる。ＳＩＭカードは、この値を、ｌｏｃａｔｉｏｎに関連付けられた値ｄａｔａと比較し、それらが同一または同等である場合、値ｓｅｅｄで応答する。デバイスは、現在のシード値をこの新しいｓｅｅｄの値で置き換える。これによって、デバイスは、所与の計算段階に達したことを証明するある値を計算した時点で、シードを置き換えることが可能である。新しいシードは、これより前には開示されず、このことはセキュリティ上のメリットがある。あるいは、新しい結果が報告されるとすぐに、この報告された値が正しかったか否かにかかわりなく、新しいシードを開示することができる。報告された値はいずれも、それが受信された時間と共に記録され、ベリファイアは、誤った結果を受け取ったかどうか、または顕著な遅延があったかどうか、後に判断することが可能である。プロキシにより送信されるレポートは、可能であれば帯域幅を節約するように、圧縮または要約することができる。

【０１８６】

マッチして、関連するシード値が配信されると、計算はトリプル（ｌｏｃａｔｉｏｎ，ｄａｔａ，ｓｅｅｄ）に関連付けられた所定のチェックポイントに達する。

【０１８７】
一部の実施形態では、ＳＩＭカードは、そのようなトリプルの長いベクトルを有しており、並びの最初のもの（ストレートな計算の場合）か、または入力にマッチするもの（外部ベリファイア・ユニットにより予測することができない分岐がある場合）か、いずれかを選択する。予測可能なｌｏｃａｔｉｏｎが用いられる場合は、値ｌｏｃａｔｉｏｎを省略することが可能である。ＳＩＭカードは、トリプルの長いリストを、それらのいずれも電話機または他の機関によって傍受される可能性なく得ることができ、これは、外部ベリファイア（または関連するプロキシ）とＳＩＭカードとの間のポイントツーポイント暗号化を用いることによって可能となる。

【０１８８】
値ｄａｔａの代わりに、ベリファイアにより開始される事象を用いることもできる。この場合、その事象がＳＩＭカードにより観測されると、関連付けられた値ｓｅｅｄが開示され、この値でデバイスでの計算が開始される。これは、クロッキングを開始するために用いることができる。また、これは、デバイスにより実施することもでき、デバイスは、計算を開始するために、例えば外部ベリファイアによって電話機に配信され得る特定のデータを提示しなければならない。

【０１８９】
シード値の一部は空とすることもでき、この場合、デバイスにシード値は送信されない。これは、通信のどのポイントに電話機が達したのかについてチェックを実行するために用いられることがある。

【０１９０】
ＳＩＭカードは、クロックされる（つまり時間を知っている）必要はなく、また、電話機上の悪質なコードによるしらみつぶし探索を不可能にするほど値“ｄａｔａ”が十分に長いのであれば、シード値を早期に得ることを目的として、敵対的環境で動作することができる。

【０１９１】
計算の終了は、上記のように、チェックポイントを持つことにより識別することができ、その後に、ＳＩＭカードは、計算時間を外部ベリファイアに報告するか、または計算の始まりから終わりまでの時間の推定値を外部ベリファイアにより計算可能にする情報を報告する。前者の場合、ローカルタイムの値の集まりをＳＩＭカードにより記録することができ、それぞれは、到達する各チェックポイントについてのものである。これは、ペア（ｌｏｃａｔｉｏｎ，ｔｉｍｅ）のリストとして記録することができ、この場合の値ｔｉｍｅはローカルタイムとすることができ、これは外部ベリファイアの時間の概念と同期している必要はない。あるいは、報告される値と関連する時刻とからなるペアを記録することができる。ＳＩＭカードは、このリストを認証し、それを外部ベリファイアに送信することができる。ベリファイアは認証が正しかったことを検証し、その後に、様々な計算タスクにどれほどの時間がかかったかを確認することができる。これを用いて、デバイスのセキュリティ態勢を推測することができる。あるいは、ＳＩＭカードが、このセキュリティについての判断を行い、その判断の認証された結果を外部ベリファイアに送信することができる。

【０１９２】
ＳＩＭカードは、時刻が分からなくても、事象を順序付けることができる。ＳＩＭカードは、ベリファイア（または、マルウェアではなくベリファイアと協働する機関）からパケットを受信する際に、デバイスにより報告されるどの事象が最も最近に生じたか判断する。これは、受信されて正しいものとして検証された値“ｄａｔａ”に相当する。この場合、値ｄａｔａは、（外部ベリファイアにより設定された、もしくはそれにより知られている何らかの計算タスクを与えられた）デバイスにより生成することが可能であり、または外部ベリファイアから伝えられた結果とすることができる。

【０１９３】
意図的に遅延されるＳＩＭカードへのレポートに対処するため、事象パケットが受信さ

れたときに、ＳＩＭカードによる外部ベリファイアへの即時の確認応答を要求することができる。これらのメッセージは、認証されるものであり、および／またはマルウェア・エージェントにより予測することができない形式になっている。これを構築する方法の例は、上記の値“ｄａｔａ”を外部ベリファイアからの事象に対応させ、関連付けられた値“ｓｅｅｄ”をＳＩＭカードから受信したら、これをデバイスにより外部ベリファイアに対して報告させることである。このとき、外部ベリファイアは、先にパケットを送信した後に、この確認応答の着信時刻を測定することができる。あるいは、デバイスは、“ｄａｔａ”に対応する値を外部ベリファイアに直接報告可能にすることができる。これは、全てのチェックポイントのうちの一部について実行することができ、または追加のチェックポイントについて実行することができる。

【０１９４】

事象Ｅが生じた後であって、他の事象が生じるよりも前に、ストリングＳにより識別されるパケットがＳＩＭカードにより受信された（さらに、ＳＩＭカードにより正しく検証された）と仮定する。この場合、ペア（Ｓ，Ｅ）がログＬに追加される。あるいはストリングＳおよびＥの識別部がログに追加される。ログは、通信の終了時にベリファイアに伝達される。あるいは、順序付け情報が外部ベリファイアに伝達される。

【０１９５】

一部の実施形態では、トリプレット（ｌｏｃａｔｉｏｎ，ｄａｔａ，ｒｅｐｏｒｔ）がベクトルに追加され、この場合の“ｒｅｐｏｒｔ”は、ベリファイアに送信されるレポートを示す値である。これは、通常のｓｅｅｄ値を用いて実現することもでき、この場合、最後に開示されるｓｅｅｄ値は、時間計測を停止させるためにデバイスがベリファイアに伝える値である。また、何らかの追加の鍵付きタスクをデバイスに実行させ、シードと引き換えに値を折り返し報告させることにより、時間計測を停止させることもできる。この場合、ストリングＳを含む次のパケットの着信によって、（目的の計算が終了した後に、計算されたと思われるサイクル数から推定することにより）時間計測が停止した時点が特定されることになる。

【０１９６】

時間計測が終了したら、ＳＩＭカードはログを暗号化（さらに、場合によっては認証）し、これをベリファイアに伝送するため、デバイスに渡す。ベリファイアは、それが正しく認証されたかどうか判断して、それを復号し、そして、Ｓのような値を含むパケットの伝送時間についての情報が所与であるとして、通信の部分的ステップの完了時間がどうであったかログから判断する。それらが電話機に最も近い基地局により送信された時刻がわかる場合があり、または、それらがネットワーク・プロキシまたはパケット自体の発信元により処理された時刻が分かる場合がある。

【０１９７】

一部の実施形態では、値Ｓは、信頼済みのビーコンにより生成され、場合によっては、デバイスで計算が開始する時点まで外部ベリファイアにより知られていない可能性がある。

【０１９８】

Ｓを含むパケットなど、パケットのタイプによっては、それらをネットワークを介して伝送する前に暗号化する必要はない。一部の実施形態では、それらは認証される。しかし、予備ログ・エントリが作成される前に、パケットが正しく認証されたかどうかについてＳＩＭカードによる検証が行われる必要はない。既に作成されたログ・エントリについて認証の検証に失敗した場合は、このログ・エントリを削除することができる。

【０１９９】

ＳＩＭカードは、半二重通信可能であり、つまり、データの受信と送信を同時に行うことはできない。ＳＩＭカードは、スレーブとして動作する。つまり、（例えば、その電源投入時など一部の特例を除いて）装着されたデバイス（電話機）へのデータの送信のみを、そのような要求があった後に行う。しかし、一部のスマートカードは、問い合わせされたことに反応してだけではなく、関連する電話機からの問い合わせの間で独立に動作する

ことができる。
【０２００】
ＳＩＭカードがマルチスレッドをサポートしている場合、（時間計測が開始するときに起動される）単純なカウントを１つのスレッドに実行させ、このカウンタを他のスレッドに提供することが可能であり、これによって、この値は、正しいデータ値が受け取られるごとに記録される。このカウンタは、データ値、関連付けられた位置、またはどのデータ値がカウンタに関連付けられているかを示す指標と共に、記憶することができる。１つのデータ値が複数回受け入れられることができないことが保証されていて、受け取られる値に確定的な順序がある場合など、状況によっては、カウンタ値を記録し、データ値またはその他の状態情報は記録しないことがある。
【０２０１】
標準的なジャバカード（ＪａｖａＣａｒｄ）など、一部のＳＩＭカードでは、ＳＩＭカード・インタフェース・デバイス（ＣＡＤ）からメッセージを受信した後にのみ、計算がサポートされる。通常、ＳＩＭカードから応答が生成されてＣＡＤに伝送されると、計算は終了する。
【０２０２】
ＳＩＭカードが、電話機からメッセージを受信する前でも、また、応答を送信した後でも、クロック・サイクルごとに（または別の確定的周期で）カウンタを進めることが可能である場合、マルチスレッドをサポートしていなくても、正確なカウントを維持することが可能である。
【０２０３】
ＳＩＭカードでは時間ベースの状態を保持することが可能である。また、ＳＩＭカードは、事象を（それが生じた時刻を含めて）認証し、そのような認証事象のリストをエクスポートすることも可能である。
【０２０４】
図１４は、デバイスの監査の一部を時間計測するプロセスの実装の一実施形態を示している。図示の例では、改良を加えたジャバカードをプロキシとして用いる。この改良によって、プロセスは要求に応答した後にアクティブのままでいることが可能となる。プロキシは、外部ベリファイアから入力および出力の値のベクトルを受け取って、ベクトルｄｕｒａｔｉｏｎを生成し、これは、実行完了時に外部ベリファイアに送信される。（プロキシと外部ベリファイアとの間の全ての通信は、認証および暗号化が行われるものと仮定される。）図示の例では、値“∞”が、クライアント上のマルウェア・エージェントによる不正の試みを示すエラーメッセージに相当する。外部ベリファイアは、ベクトルｄｕｒａｔｉｏｎを受信すると、その中の値が全て、正常に完了したことを意味する厳しい制限範囲内にあるかどうか判断し、そうであると言える場合にのみ、クライアントが安全な状態にあると結論付ける。
【０２０５】
時間計測にプロキシを用いる追加の方法
【０２０６】
ＳＩＭまたはこれに類するハードウェアをプロキシとして用いることに加えて、安全な状態であると考えられる他のデバイスをプロキシとして用いることも可能である。最初に（電話機など）１つのデバイスが安全であることを検証することによってセキュリティをブートし、その後そのデバイスを用いて、第２のデバイスのセキュリティを検証するプロセスでローカルタイムの計測および他の検証タスクを実行することが可能である。
【０２０７】
デバイスは、様々に異なるタイプのものとすることができる。例えば、外部ベリファイアの要求に応じて、ＳＩＭカード、ローカルな携帯電話の中継塔、またはローカル・コンピュータを用いて、電話機のセキュリティ検証を支援してもよい。この外部ベリファイアは、上記のペア（ｄａｔａ，ｓｅｅｄ）など、検証で用いられるデータの事前に用意された部分を持つことが可能である。また、この事前計算は、第三機関により実行することも

可能である。様々な実施形態において、シードは、プロキシによって生成される。一例では、プロキシとデバイスの両方が、加速度計または光起電センサなどのセンサを備える。デバイスとプロキシの両方が（例えば、一緒に握られて、一緒に振られることにより）同じ現象を観測して、シードを計算する。この場合、デバイスは観測されたシードを使用し、プロキシは（それが独立に経験した）シードを外部ベリファイアに送信する。

【０２０８】

第１のデバイスが安全な状態にあると判断されると直ちに、それを、自動車のインフォテインメント・システム、別の電話機、ネットブック、ラップトップまたはデスクトップまたはその他のデバイスなど、第２のデバイスのセキュリティ評価における時間計測またはその他の支援に用いることができる。このセキュリティ評価は、（例えば、計算タスクを実行する時間に基づく）同じタイプのものとすることができ、または第１のものの上でブートされる代替のセキュリティ評価方法とすることができる。同様に、第１のセキュリティ評価を異なるタイプのものとすることができ、一方、後のセキュリティ評価は、第１のものの上でブートすることができ、これは、計時ベースのアプローチを用いるものであってもよい。

【０２０９】

一部の実施形態では、最初に知られる“安全な”デバイスを用いて、シード値のセットの集まりが生成され、これらは後の時点で他のデバイスによって使い尽くされるものである。そのようなセットを、（例えば、ＰＫＩを用いて、または値のピア・ベースの認証により）認証することが可能である。また、外部ベリファイアは、オンライン・プロキシの支援を得られる場合は、オフラインで動作することが可能である。例えば、外部ベリファイアは、暗号化および認証されたデータを、監査プロセスよりずっと前の時点で送信することができる。いくつかのそのようなトランスクリプトを一度に送信することができる。それらは、それらを保持することができるプロキシに送られるか、または、監査されるデバイスもしくはそのプロキシに送られて、そこで必要になるまで保持されることになるか、いずれかとすることができる。メモリ印刷の結果としてプロキシにより生成されるトランスクリプトも、やはりデバイスによってバッファリングされる場合があり、後に、それらが要求されたかどうかにかかわらず、または通信チャネルが使用できるときに、外部ベリファイアに送られる。一部の実施形態では、全てのレコードは、それらが認証および場合によってさらに暗号化される前に、タイムスタンプおよびシリアル番号でマークアップされる。

【０２１０】

これを、小ノードのネットワークで実施することが可能であり、その場合、一部のノードは、事前に信頼済みであるか、または安全であると評価されているか、どちらかである。これらのノードは、その後、他のノードのセキュリティ評価を支援するために用いられる。これは、再帰的アプローチである可能性があり、また、循環的とすることができ、つまり、先に信頼済みとなって他のデバイスのセキュリティ評価に用いられるデバイスは、後に、これらのデバイスの一部または他の方法で安全であると評価されたデバイスによって検証されることがある。外部ベリファイアは環境に含めることもでき、検証事象の連鎖の開始を支援し、どのノードをいつ誰により検証すべきかのスケジューリングを支援することがある。

【０２１１】

圧縮アクセステーブルの利用

【０２１２】

一部の実施形態において、メモリアクセスの位置はベクトル“ｌｏｃａｔｉｏｎ”の内容により決定され、その内容は、空きＲＡＭの全てのセルの順列に対応している。このベクトルは、空きＲＡＭに保持された場合、そのすべてを占めることができる。それは、一部の実施形態では、（フラッシュメモリなど）二次記憶装置に記憶されており、必要に応じて部分的にスワップインされる。擬似ランダム・アクセス順序を維持しながらメインループでの計算量を最小限に抑える代替アプローチについて、以下で説明する。

【０２１３】
２つのベクトル、ｌｏｃａｔｉｏｎＨとｌｏｃａｔｉｏｎＬを考える。この両方が、それぞれ部分的なメモリアクセス位置の順列を含むベクトルである。ここで、実際のメモリアクセス位置は、２つの部分的な位置の組み合わせであり、例えば、１つの要素ｌｏｃａｔｉｏｎＨのビットが、１つの要素ｌｏｃａｔｉｏｎＬのビットと連結されたものである。ここで、要素ｌｏｃａｔｉｏｎＨは上位ビットを含み、要素ｌｏｃａｔｉｏｎＬは下位ビットを含むと仮定される。これらの要素は、同一または異なるサイズとすることができるが、組み合わせたときには、１つのメモリ位置をアドレス指定するサイズとなる。それぞれが範囲内の全ての可能な要素を含んでいれば、全ての組み合わせの集まりは、全てのメモリアドレスの集まりに相当することになる。（ここから、組み合わせ結果を閾値と比較することにより、空きＲＡＭ内にないものを取り除くことができ、その結果が閾値より低い場合は捨てられる。）この表現は、記憶のためにアドレス指定される空間の大きさの平方根のスペースしか取らない。３つのコンポーネントを用いることが可能であり、この場合、これらはアドレス指定される空間の三乗根のスペースを取る。同様に多数のコンポーネントを用いることも可能である。結合関数の一例は、連結である。
【０２１４】
一部の実施形態では、ベクトルの要素のアクセス順序は、圧倒的な可能性で全ての組み合わせが使用されることになるように保証される幾何学的パターンに従う。メモリアクセスに多くのパターンが与えられているとして、１つのベクトル内の１つの同じ項目の複数のアクセスを持たないことは、これによって攻撃者にとっての予測不可能性の程度が減少するけれども、有益となり得る。全てのアクセスが行われることを保証すると同時にメモリへのアクセス総数を制限することは有益であるにもかかわらず、１つの組み合わせを複数回カバーすることが可能であるが、しかし、可能性はごくわずかである。
【０２１５】
位置ｘでｌｏｃａｔｉｏｎＨベクトルに、位置ｙでｌｏｃａｔｉｏｎＬベクトルにアクセスし、対角線に沿ってｘ‐ｙ位置にアクセスすることが可能である。ここで、最初のシーケンスは、位置（ｘ，ｙ）＝（０，０）で開始させることができ、その後、ループの繰り返しごとに、ｘとｙが両方とも同時に１ずつ増加される。１つの座標がベクトルの大きさを超えて増加されたら、その座標は再び０に設定される。そして、位置が再び（０，０）になると、これを変更して、位置（ｘ，ｙ）＝（１，０）で開始させることができ、その後、インクリメントのシーケンスが繰り返され、再び（１，０）になると、これが（２，０）に変更される。これは、メモリアクセスの場所ではない。これは、メモリアクセスを行う場所を記述するベクトル内での位置である。
【０２１６】
また、それ以外に、場所の要素からなるベクトルを持つことで、どのセルにアクセスするかについての記述を圧縮することも可能であり、この場合、そのような各場所は、アドレスの一部のみを記述し、アドレスの残りのビットは、別の方法で計算されるか、または計算時のプログラムの状態から推測される。さらに、これらの２つのアプローチを組み合わせることができ、また、少なくとも部分的に事前に生成されるアクセス場所のさらに別の関連記述と組み合わせることができる。
【０２１７】
時間計測に関する追加情報
【０２１８】
様々な計算において、計算および部分的計算の時間計測は、次のように行われる。（Ａ）外部ベリファイアまたはそのプロキシから監査部に必要な値が全て提供されると、タイマーが開始される。これらの値は、一般的に値ｓｅｅｄを含んでいる。（Ｂ）監査部が正しい値“ｓｔａｔｅ”を外部ベリファイアまたはそのプロキシに送信すると、タイマーが停止される（そして、開始してからの時間が記録される）。
【０２１９】
古いものが終了すると、新しい時間間隔を直ちに開始させることが可能である（この場

合、開始は上記のステップＡにより表され、終了はステップＢにより表される）。また、これらの時間間隔の間に“休み時間”を組み込むことも可能である。これらの休み時間の間は、計算の時間計測が行われなくてもよく、アルゴリズムは、外部機関との通信、二次記憶装置の読み取りもしくは書き込み、または他の機能などのメンテナンス・ルーチンを実行することがある。休み時間は、アルゴリズムが次の時間間隔の開始（例えば、ステップＡ）を要求すると、終了させることができる。これを行うことができる一つの方法は、外部ベリファイアまたはそのプロキシに次の時間間隔の開始を信号で伝えることによる。あるいは、外部ベリファイアまたはそのプロキシによって新しい時間間隔の開始を選択することにより、行うことができる。

【０２２０】

また、休み時間を標準的な時間間隔として組み込むことも可能であり、その長さは、監査されるデバイスのセキュリティ態勢についての最終的な判断にとって重要ではない。

【０２２１】

擬似ランダム・アクセス

【０２２２】

一部の実施形態において、監査プロセスの一部として実行される選択的な読み取りは、擬似ランダム順序でのアクセスによるアクセス位置の読み取りおよび書き込みのシーケンスによって達成される。擬似ランダム・アクセスを用いる代替の実施形態について以下で説明する。まず、メモリ充填の一例についての説明を行う。そして次に、周期的な時間計測の一例についての説明を行う。

【０２２３】

高速メモリの充填

【０２２４】

次のメモリ印刷機能を用いて空きＲＡＭを埋めることができる。また、これを他のタイプの高速メモリの充填に用いることも、そのような他のタイプのメモリがアクセス時間の面でＲＡＭと同等である場合には可能である。擬似ランダム・シーケンスが、擬似ランダムな順序で空きＲＡＭにＸＯＲされる。その後、ＲＡＭの内容全体の鍵付きハッシュが計算される。ＲＡＭはブロックおよびページを使用していないが、それでも“仮想の”ブロックおよびページに分割することができ、これらはフラッシュメモリのそれに対応している。フラッシュメモリの連続するチャンクは、ページまたはブロックでアクセスされない。これによって、フラッシュメモリではアクセスが遅くなるが、ＲＡＭでは依然として高速である。

【０２２５】

擬似ランダム・ストリングで空きＲＡＭを埋めるために、２つの主なステップがある。最初に、セットアップ機能が実行される。これにより、ベリファイアから得られる擬似ランダム値生成のためのシードを用いて、メモリ印刷機能により行われるメモリアクセスのランダムな順序が決定される。テーブルはフラッシュメモリに記憶され、セットアップ機能により使用されるプログラム領域は、セットアップが完了した後にクリアされる。二番目に、メモリ印刷機能を用いて空きＲＡＭを全て充填する。その実行は、始まりから終わりまでと、それより短い間隔の両方で、時間計測される。

【０２２６】

ネットワーク遅延への対処

感染によって引き起こされる遅延は、内部配線により；ＵＳＢなど標準のネットワークポートにより；有線インタフェースを介して；Ｗｉ‐Ｆｉネットワークを介して；ＬＡＮを介して；インターネットを介して；パケット交換ネットワークを介して；通信ネットワークを介して、またはこれらの組み合わせによりクライアント・デバイスに接続されるデバイスから、測定することができる。これらの通信媒体の一部は、遅延および変動を取り込むことがあり、これらは、統計的手法を用いて測定値から分離することができる。

【０２２７】

検証は、監査されるデバイスに、ケーブル、ＬＡＮ、ＷＡＮ、ブルートゥース、Ｗｉ‐

Ｆｉ、インターネット、他のネットワーク、またはネットワークの組み合わせを用いて接続されたデバイスによって行われる。検証は、受信した結果を、計算した結果と比較することにより行われ、それ（および、それの前のシーケンス）が適切な時間の範囲内で受信されたことが確認される。これらの通信媒体は全て、待ち時間が生じることがあり、また、一部のものはパケットが欠落する場合がある。

【０２２８】

“良い”事象のときは、１０単位時間プラス１～５（一般的なネットワーク変動の場合）の時間がかかると仮定する。

【０２２９】

この場合、“悪い”事象は、１５単位時間に加えて、ネットワーク変動の１～５の時間がかかると仮定する。

【０２３０】

これらの時間での途中結果の受信を考える。

【０２３１】

シーケンスａ：０，１２，２５，（パケット欠落），５０―このシーケンスは、欠落パケットがあるにもかかわらず、最後の途中結果が欠落パケットを“保証”しているので、良い可能性が高い。

【０２３２】

シーケンスｂ：０，１１，３０，３５，５０―このシーケンスは、２番目と３番目のパケットの間に大きな遅延があるにもかかわらず、４番目のパケットの受信が“早すぎる”ので、良い可能性が高い。

【０２３３】

シーケンスｃ：０，１１，３０，４５，５７―このシーケンスは、第２のパケットの後に大きな遅延があり、この遅延を説明する事象がないので、悪い可能性が高い。

【０２３４】

上記の実施形態は、明確な理解を目的として、ある程度詳細に記載したが、本発明は、提示した詳細に限定されるものではない。本発明を実施する多くの代替の方法がある。開示された実施形態は、例示であって、限定するものではない。

本発明は、たとえば、以下のような態様で実現することもできる。

適用例１：
物理メモリと、
プロセッサと、を備えるシステムであって、
前記プロセッサは、
ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、さらに初期化情報を受け取り、
前記物理メモリを選択的に読み取って、少なくとも１つの結果を決定し、
前記結果をベリファイアに提供するように構成されている、システム。

適用例２：
適用例１に記載のシステムであって、
前記初期化情報はシード値を含む、システム。

適用例３：
適用例１のシステムであって、
前記初期化情報はステップ値を含む、システム。

適用例４：
適用例１のシステムであって、
前記初期化情報はプロキシから受け取られる、システム。

適用例５：
適用例１のシステムであって、
前記初期化情報は外部ベリファイアから受け取られる、システム。

適用例６：
適用例１のシステムであって、
前記初期化情報は現象の表現である、システム。

適用例７：
適用例６のシステムであって、
前記初期化情報は、前記システムが備える加速度計から受け取られる、システム。

適用例８：
適用例６のシステムであって、
前記プロセッサは、少なくとも部分的に、現象を測定することにより、前記初期化情報を受け取るように構成されている、システム。

適用例９：
適用例１のシステムであって、
前記プロセッサは、さらに、前記物理メモリの上書きされる部分を関数に従って決定するように構成されている、システム。

適用例１０：
適用例１のシステムであって、
前記プロセッサは、さらに、関数に従って前記物理メモリの一部に書き込みを行うように構成されている、システム。

適用例１１：
適用例１のシステムであって、
前記プロセッサは、ステップ・カウンタに少なくとも部分的に基づいて、前記物理メモリを選択的に読み取るように構成されている、システム。

適用例１２：
適用例１１のシステムであって、
前記ステップ・カウンタは、前記物理メモリのサイズの測定値に対して、互いに素である、システム。

適用例１３：
適用例１１のシステムであって、
複数のプロセッサをさらに備え、
前記ステップ・カウンタは、前記物理メモリのサイズの測定値に前記プロセッサの数を乗じたものに対して、互いに素である、システム。

適用例１４：
適用例１のシステムであって、
前記プロセッサは、恒等関数ではない関数であって、前記物理メモリの一部を入力として受け取る関数を少なくとも部分的に用いることにより、前記結果を決定するように構成されている、システム。

適用例１５：
　適用例１のシステムであって、
　前記プロセッサは、アキュムレータを少なくとも部分的に用いることにより、前記結果を決定するように構成されている、システム。

適用例１６：
　適用例１５のシステムであって、
　前記アキュムレータは非線形関数を含んでいる、システム。

適用例１７：
　適用例１５のシステムであって、
　前記アキュムレータは、ＸＯＲおよびローテートを用いる、システム。

適用例１８：
　適用例１のシステムであって、
　前記プロセッサは、少なくとも部分的に、プロキシに情報を提供することにより、前記結果を前記ベリファイアに提供するように構成されている、システム。

適用例１９：
　ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、さらに初期化情報を受け取ることと、
　物理メモリを選択的に読み取って、少なくとも１つの結果を決定することと、
　前記結果をベリファイアに提供することと、を含む方法。

適用例２０：
　コンピュータ読み取り可能な記憶媒体において実現されるコンピュータプログラム・プロダクトであって、
　ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、さらに初期化情報を受け取り、
　物理メモリを選択的に読み取って、少なくとも１つの結果を決定し、
　前記結果をベリファイアに提供する、
ためのコンピュータ命令を含む、コンピュータプログラム・プロダクト。

適用例２１：
　物理メモリと、
　プロセッサと、を備え、
　前記プロセッサは、
　ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、
　前記物理メモリに対して変更のシーケンスを実行し、
　結果をベリファイアに提供するように構成されている、システム。

適用例２２：
　適用例２１のシステムであって、
　前記ベリファイアは、物理メモリ変更の期待されるシーケンスが前記プロセッサにより実行されたかどうかを、受信した前記結果に少なくとも部分的に基づいて判断するように構成されている、システム。

適用例２３：
　適用例２２のシステムであって、
　前記判断は、前記変更のシーケンスの実行に関連するタイミング情報に少なくとも部分

的に基づいている、システム。

適用例２４：
適用例２２のシステムであって、
前記判断は、デバイスから受け取る結果の正確さに少なくとも部分的に基づいている、システム。

適用例２５：
適用例２１のシステムであって、
前記ベリファイアは、デバイス上で前記プロセッサと併置されている、システム。

適用例２６：
適用例２５のシステムであって、
前記ベリファイアは、加入者識別モジュールに含まれている、システム。

適用例２７：
適用例２１のシステムであって、
前記ベリファイアは、ネットワーク接続を介して前記プロセッサと通信する、システム。

適用例２８：
適用例２１のシステムであって、
前記プロセッサは、さらに、前記ベリファイアからシードを取得するように構成されている、システム。

適用例２９：
適用例２１のシステムであって、
前記変更のシーケンスは、前記ベリファイアから取得される入力に少なくとも部分的に依存する、システム。

適用例３０：
適用例２１のシステムであって、
前記１つまたは複数のハードウェア・パラメータは、メモリの容量である、システム。

適用例３１：
適用例２１のシステムであって、
前記１つまたは複数のハードウェア・パラメータは、メモリの速度である、システム。

適用例３２：
適用例２１のシステムであって、
前記物理メモリ内にアクティブである回避的ソフトウェアはないと判断されたら、前記プロセッサは、スキャンを実行するように構成されている、システム。

適用例３３：
適用例３２のシステムであって、
前記スキャンは、ライセンスを受けていないソフトウェアのスキャンを含む、システム。

適用例３４：
適用例２１のシステムであって、

前記メモリは秘密データを含み、
前記ベリファイアに結果を提供することによって、前記秘密データが漏洩することはない、システム。

適用例３５：
適用例２１のシステムであって、
前記プロセッサは、携帯電話機の中に含まれている、システム。

適用例３６：
ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取ることと、
物理メモリに対して変更のシーケンスを実行することと、
結果をベリファイアに提供することと、を含む方法。

適用例３７：
適用例３６の方法であって、
前記ベリファイアは、物理メモリ変更の期待されるシーケンスが実行されたかどうかを、受信した前記結果に少なくとも部分的に基づいて判断するように構成されている、方法。

適用例３８：
コンピュータ読み取り可能な記憶媒体において実現されるコンピュータプログラム・プロダクトであって、
ハードウェア構成に対応する１つまたは複数のハードウェア・パラメータを受け取り、
物理メモリに対して変更のシーケンスを実行し、
結果をベリファイアに提供する、
ためのコンピュータ命令を含む、コンピュータプログラム・プロダクト。

適用例３９：
プロセッサと、
前記プロセッサに結合され、前記プロセッサに命令を提供するように構成されたメモリと、を備えるシステムであって、
前記プロセッサは、
デバイス上のメモリへの変更のシーケンスの前記デバイスによる実行に関連する１つまたは複数の結果を、前記デバイスから受け取り、
前記デバイス上の前記メモリ内には回避的ソフトウェアがないことを、前記１つまたは複数の結果が示していることを確認し、
前記確認が行われた後に、前記デバイスのスキャンを開始する、ように構成されている、システム。

適用例４０：
プロセッサと、
前記プロセッサに結合され、前記プロセッサに命令を提供するように構成されたメモリと、を備えるシステムであって、
前記プロセッサは、
デバイスが結果を決定するように要求を開始させ、
前記デバイスから前記結果を受け取り、
前記結果、および前記結果を前記デバイスが提供するのに要した時間の長さに少なくとも部分的に基づいて、前記デバイスのセキュリティ状態を確認する、ように構成されている、システム。

【図１】

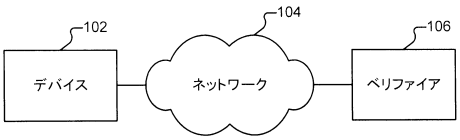

FIG. 1

【図２】

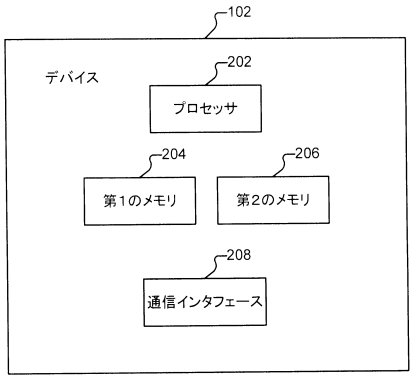

FIG. 2

【図３】

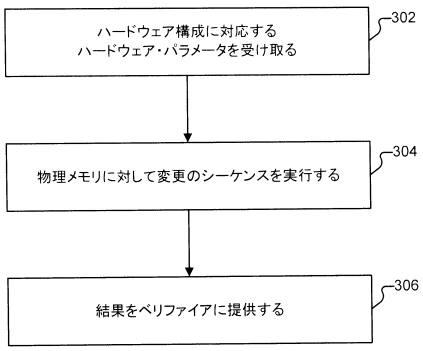

FIG. 3

【図４】

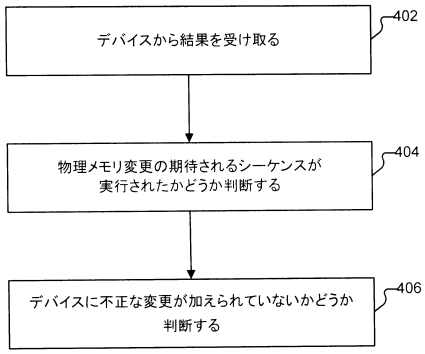

FIG. 4

【図５Ｂ】

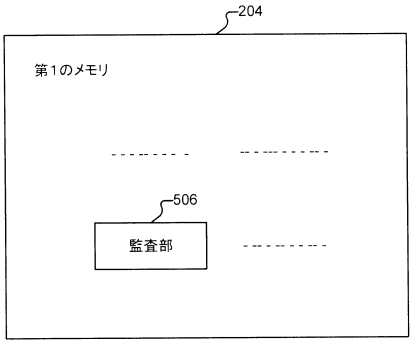

FIG. 5B

【図５Ａ】

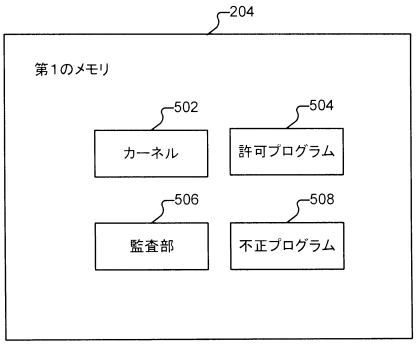

FIG. 5A

【図６】

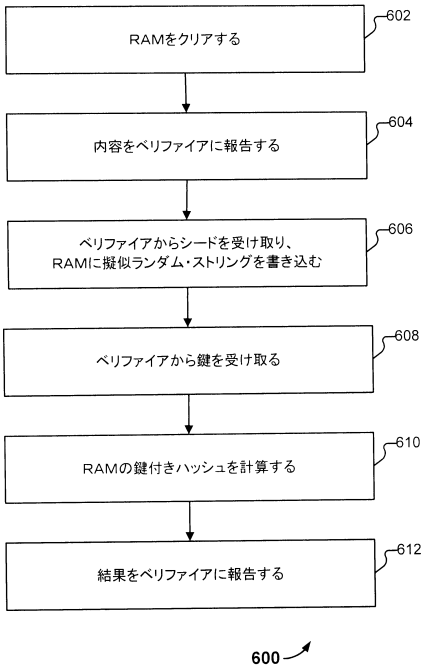

FIG. 6

【図７】

```
for i:=1 to rounds
  for k:=0 to chunks_per_block -1
    permutation[1  number_blocks] := get_permutation
    for j:= 1 to number_blocks
      modify_memory(permutation[j]*chunks_per_block+k, next_string_chunk)
```

FIG. 7

【図８】

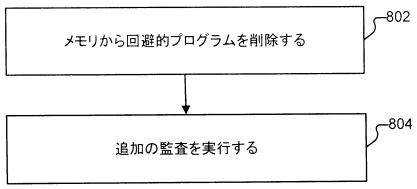

FIG. 8

【図９】

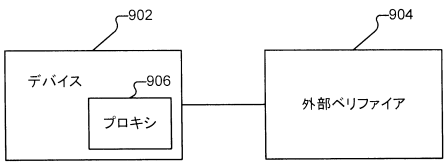

FIG. 9

【図１０】

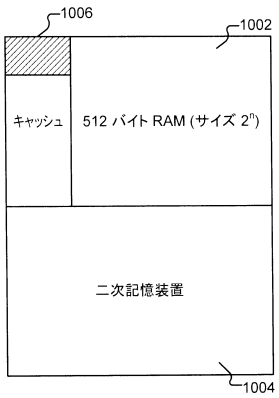

FIG. 10

【図１１】

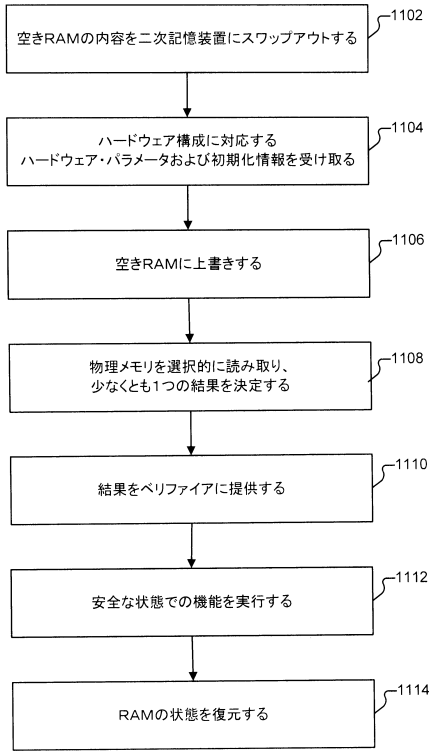

FIG. 11

【図１２】

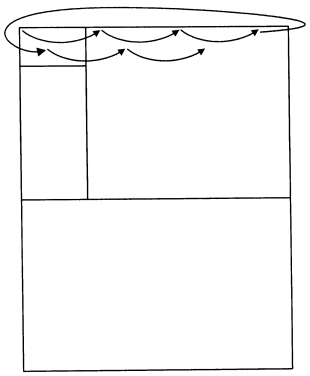

FIG. 12

【図１３】

$state \leftarrow 0$
$location \leftarrow 0$
repeat $(s_{RAM}/period/\alpha)$ times:
  receive value key
  $state \leftarrow state$ XOR key
  repeat $period$ times:
    $data \leftarrow \mathrm{RAM}[location]$ % read a cache line
    $state \leftarrow \mathrm{ROR}(state$ XOR $data)$ % accumulation
    $data \leftarrow \mathrm{RAM}[location + 1]$ % read from cache
    $state \leftarrow \mathrm{ROR}(state$ XOR $data)$ % accumulation
    ...
    $data \leftarrow \mathrm{RAM}[location + \alpha - 1]$ % read from cache
    $state \leftarrow \mathrm{ROR}(state$ XOR $data)$ % accumulation
    $location \leftarrow (location + step) \bmod s_{RAM}$ % next location?
  report value $state$

FIG. 13

【図１４】

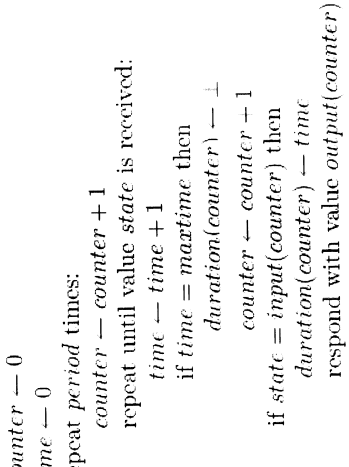
counter ← 0
time ← 0
repeat period times:
counter ← counter + 1
repeat until value state is received:
time ← time + 1
if time = maxtime then
duration(counter) ← ⊥
counter ← counter − 1
if state = input(counter) then
duration(counter) ← time
respond with value output(counter)
else
duration(counter) ← ⊥

FIG. 14

フロントページの続き

(31)優先権主張番号 61/234,604
(32)優先日 平成21年8月17日(2009.8.17)
(33)優先権主張国 米国(US)
(31)優先権主張番号 61/257,043
(32)優先日 平成21年11月2日(2009.11.2)
(33)優先権主張国 米国(US)
(31)優先権主張番号 12/580,891
(32)優先日 平成21年10月16日(2009.10.16)
(33)優先権主張国 米国(US)

(72)発明者 ヨハンソン・カールーアンドレス・アール.
アメリカ合衆国 カリフォルニア州９４０４１ マウンテン・ビュー，マリポサ・アベニュー，５９０

審査官 戸島 弘詩

(56)参考文献 特開２００６－１２７５２１（ＪＰ，Ａ）
特表２００８－５１４１５０（ＪＰ，Ａ）
国際公開第２００８／０１０８５３（WO，Ａ１）
特表２００９－５４３１８６（ＪＰ，Ａ）
国際公開第０１／０３１４２１（WO，Ａ１）
米国特許出願公開第２００８／０２２９４１５（ＵＳ，Ａ１）
特表２００８－５０１９６４（ＪＰ，Ａ）
米国特許出願公開第２００８／０１８４３７１（ＵＳ，Ａ１）
国際公開第２００８／０４８６６５（WO，Ａ１）
国際公開第２００９／０３１０６５（WO，Ａ１）
米国特許出願公開第２００８／００３４４０６（ＵＳ，Ａ１）
特表２００９－５４４０９８（ＪＰ，Ａ）
Arvind Seshadri et.al., SWATT: SoftWare-based ATTestation for Embedded Devices, Proc. of the IEEE Security & Privacy Conference, IEEE, ２００４年 5月３１日, [平成２６年２月１４日検索], ＵＲＬ, http://paramecium.us/~leendert/publications/swatt.pdf
西原 清一，Ｃで学ぶ データ構造とアルゴリズム，株式会社オーム社，２００８年 1月２０日，第1版，第１８６－１８７頁

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ２１／５６
Ｇ０６Ｆ１１／２２－１１／２６
Ｇ０６Ｆ１１／２８－１１／３４