(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2017-139793
(P2017-139793A)
(43)公開日　平成29年8月10日(2017.8.10)

(51)Int.Cl.		F I			テーマコード（参考）
H04L 9/32	(2006.01)	H04L	9/00	675B	5J104
G09C 1/00	(2006.01)	G09C	1/00	640E	
G06F 21/44	(2013.01)	G06F	21/44		
G06F 21/64	(2013.01)	G06F	21/64		

審査請求 有　請求項の数 1　ＯＬ　（全 16 頁）

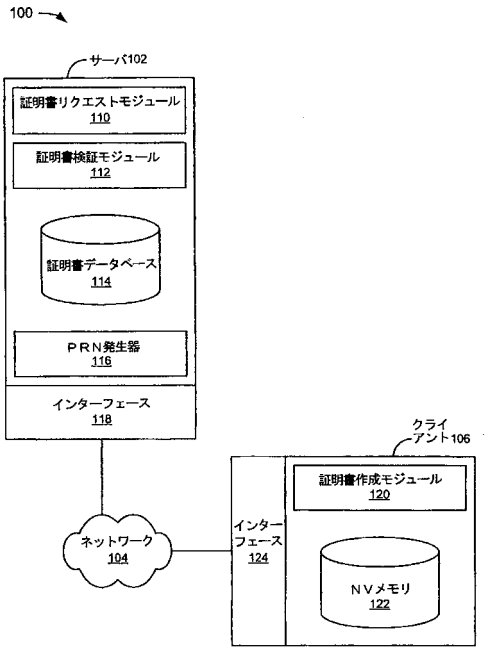
(21)出願番号　特願2017-60504 (P2017-60504)
(22)出願日　平成29年3月27日 (2017.3.27)
(62)分割の表示　特願2015-149224 (P2015-149224)の分割
原出願日　平成19年9月13日 (2007.9.13)
(31)優先権主張番号　60/857,840
(32)優先日　平成18年11月9日 (2006.11.9)
(33)優先権主張国　米国 (US)
(31)優先権主張番号　11/601,323
(32)優先日　平成18年11月16日 (2006.11.16)
(33)優先権主張国　米国 (US)

(71)出願人　512168733
エーサー・クラウド・テクノロジイ・インコーポレイテッド
アメリカ合衆国　カリフォルニア州　９４０４１－２９２０　マウンテン　ビュー　スイート　１００　ビラ　ストリート　１２００
(74)代理人　100098394
弁理士　山川　茂樹
(74)代理人　100064621
弁理士　山川　政樹
(72)発明者　スリニバサン，プラミラ
アメリカ合州国，カリフォルニア州　９４３０３，パロ　アルト，チャニング　アベニュー　１８５３

最終頁に続く

(54)【発明の名称】サーバ

(57)【要約】
【課題】セキュアなシステムを提供する。
【解決手段】数発生器が第１の数を生成し、
証明書リクエストモジュールは、デバイス証明書のリクエストを生成し、前記第１の数と前記デバイス証明書のリクエストはインターフェースを介してクライアントに送られ、
前記第１の数と、この第１の数を使用して前記クライアントにより生成される第２の署名と、デバイス証明書および該第１の数を含むレスポンスが前記インターフェースで受け取られ、
証明書検証モジュールは前記デバイス証明書と第２の署名を有効化し、前記数発生器が発生した第１の数と前記クライアントからの第１の数が一致しているか否かを検証する。
【選択図】　図１



【特許請求の範囲】
【請求項１】
数発生器と、
証明書リクエストモジュールと、
証明書検証モジュールと、
前記数発生器、前記証明書リクエストモジュール、及び前記証明書検証モジュールに接続するインターフェースとを備えるとともに、不揮発性メモリを有したクライアント装置を認証するためのサーバであって
動作において、
前記数発生器が第１の数を生成し、
前記証明書リクエストモジュールは、デバイス証明書のリクエストを生成し、
前記第１の数と前記デバイス証明書のリクエストは前記インターフェースを介してクライアント装置に送られ、
第２の数と、
前記クライアント装置において該クライアント装置の製造中にプログラムされたプライベートキーと前記サーバの数発生器が作成した第１の数を使用して生成される第２の署名と、
前記クライアント装置の装置ＩＤ、発行者ＩＤ、前記プライベートキーの関数である公開鍵及び前記クライアント装置の不揮発性メモリから読み出された第１の署名の関数として生成されたデバイス証明書とを含むレスポンスが前記インターフェースで受け取られ、
前記証明書検証モジュールは前記デバイス証明書と第２の署名とを認証し、前記サーバの数発生器が発生した第１の数と前記クライアント装置からの第２の数が一致しているか否かを検証する
ことを特徴とするサーバ。
【発明の詳細な説明】
【技術分野】
【０００１】
セキュアプロセッサは通常、ＩＤ、及び／又は格納された秘密鍵を含む。セキュリティレベルを高めるために、ある量(quantities)の秘密鍵などが、チップ内不揮発性メモリにプログラミングされて、セキュアプロセッサを作ることがある。ＩＤ及び秘密鍵のプログラミングは、チップのセキュア製造プロセスにおいて行われる。それぞれのＩＤは固有のものであり、プライベートキーも又固有のものである。これらの量(these quantities)は、デジタル権利管理や他のセキュリティ関連アプリケーションを実行するため、装置上のアプリケーションで使用される。通常、チップは、ネットワーク・プロトコル、秘密鍵などにおいて、使い捨てデータとして使用するために、暗号学的に強力な乱数を生成するメカニズムを含む。
【背景技術】
【０００２】
デジタル権利管理を実行するために使用される典型的なインフラにおいて、サーバは、装置に対する権利を有効にするため、デジタル的に署名されたチケットを供給するのに使用される。そのようなチケットは、装置にチケットを結びつけるために、かかる装置のアイデンティティ、及び／又は秘密鍵メカニズムを使用する。それぞれの装置ＩＤ／鍵の独自性を確実にするため、サーバは通常、セキュア・データベースを使って、製造された各チップに対応するＩＤ（及び／又は署名された証明書）を格納している。これらの証明書は、チップにプログラミングされた各秘密鍵（（プライベートキー、公開鍵）ペアのプライベートキー）に対応する公開鍵を含んでいる。データベースに証明書を投入するためには、データベースに関連するインフラを確実に製造プロセスに結び付け、製造されたチップとデータベース内の証明書の間で１対１対応を維持しなければならない。
【０００３】

上述したこれら関連技術の例や制限事項は、あくまで説明に役立つものとして例証したものであって排他的ではない。関連技術の他の制限事項は、当該技術者が本願明細書を読み、図面を検討することにより明らかになるであろう。

【発明の概要】

【発明が解決しようとする課題】

【０００４】

以下の実施形態とその特徴は、あくまで例証的であって範囲を制限しないことを意図したシステム、ツール、および方法に関連して記載、図示される。種々の実施形態では、上述した問題の１つ又はそれ以上が低減又は解消され、一方では他の実施形態がその他の改善に向けられる。

【０００５】

改善されたセキュア・プログラミング技術は、セキュアデバイスによってサポートされる典型的なセキュア・アプリケーションを可能とすると同時に、オンチップ秘密不揮発性メモリにプログラミングされるビットサイズを減少させることを含む。また、改善されたセキュア・プログラミング技術は、システムの製造プロセスを簡素化することを含む。一実施形態において、秘密事項のプログラミングは、オンチッププログラミングに対し分離されており、特にシステム統合やインフラ設定のプロセスからは分離される。

【課題を解決するための手段】

【０００６】

セキュア・プログラミング技術は、チップの製造を、チケットを取得するために、チケットサーバに接続する後処理から分離させることを含む。この技術に沿った方法は、チケットを受け取るための如何なる通信に先だって、装置から（製造）サーバの署名が入った証明書を送ることを含んでも良い。この方法は、データベースを投入し、例えばちょうどチケットサービスが必要となる時など、後のチケットサービス実行を容易にすることを含んでも良い。

【０００７】

本技術に沿った装置は、製造プロセスにおいて、証明書をプライベートキーと共に格納するためのチップ内不揮発性メモリを含んでも良い。プライベートキーは、楕円カーブをベースとするプライベートキーであっても、またそうでなくとも良い。楕円カーブの暗号ベースの鍵の長所は、相対的な暗号強度にしては、多くの種類の鍵より小さいことである。更に、楕円曲線アルゴリズムを使用し、無作為のプライベートキーを保存すると共に、ランタイム計算によって公開鍵を算出することが可能である。

【０００８】

有利な点としては、特にオンチップ・リアルエステートを考慮して、圧縮された証明書を不揮発性メモリ内に設けることが可能なことがある。（デバイス証明書を保存するために必要とされるものより）より小さなデータセットを使用して、かかる装置はデバイス上にダイナミックに証明書を作成し、それを要求するアプリケーションに提供する。デバイス証明書を複数回作っても、また作らなくとも良い。例えば、一旦デバイス証明書を作成し、更なる使用に備えてシステムの外部記憶装置に格納するようにしても良い。このことは証明書が公知データであるため、特に不安定なことではない（セキュアでない訳ではない）。

【０００９】

本技術によって構築された装置は、他の分野においても適用性を持つ可能性がある。例えば、ピア又は、第１のデバイス証明書を必要とする如何なるアプリケーションに対しても装置を認証することができる。またこれとは別に、不揮発性メモリにプログラミングするためのセキュアな製造プロセスを使い、不揮発性メモリは、装置のためのセキュアな乱数発生器を含んでも良い。

【図面の簡単な説明】

【００１０】

本発明の実施形態を図面に示す。しかしながら、ここに示した実施形態と図面は、本発

明を制限するものというよりむしろ説明するためのものであって、本発明の例を提供するものである。
【図１】サーバにおいてクライアントを認証するためのシステム例を示す図である。
【図２】このシステムにおける使用に適切な装置の電源オンからオフの方法例を示すフローチャートである。
【図３】１回のみデバイス証明書を作成するための方法例を示すフローチャートである。
【図４】図１乃至図３を参照して説明した技術を実装の形態に適したコンピュータ・システムを示す図である。
【図５】図１乃至図３を参照して説明した技術の実装の形態に適したセキュアシステム例を示す図である。
【図６】セキュアデバイス製造方法の一例を示すフローチャートである。
【図７】セキュア証明書の作成方法の一例を示すフローチャートである。
【発明を実施するための形態】
【００１１】
以下の説明では、本発明の実施形態の十分な理解を与えるために幾つかの特定の具体的な構成が提示される。しかし、当業者であれば、１つ以上のこれら特定の具体的な構成がなくとも、あるいは他の構成部などとの組み合わせにより、本発明を実施することが可能であることが理解されるであろう。また別の例では、様々な実施形態における本発明の側面を覆い隠すことがないように、周知の具体的構成は、図示または詳細に説明されていない。
【００１２】
図１は、サーバにおいてクライアントを認証する(validating)ためのシステム１００の一例を示している。図１の例でシステム１００は、サーバ１０２、ネットワーク１０４、及びクライアント１０６を含む。サーバ１０２は、証明書リクエストモジュール１１０、証明書検証モジュール１１２、証明書データベース１１４、擬似乱数（ＰＲＮ）発生器１１６、及びインターフェース１１８を含む。クライアント１０６は、証明書作成モジュール１２０、不揮発性（ＮＶ）メモリ１２２、及びインターフェース１２４を含む。
【００１３】
サーバ１０２は、如何なる適用可能な公知の（又は手頃な）コンピュータであっても良い。ネットワーク１０４は、非限定的な例として、インターネットを含む如何なる通信ネットワークでも良い。クライアント１０６は、セキュア記憶装置（技術的に安全性が保証された記憶装置）を有する、如何なる適用可能な公知の（又は手頃な）コンピュータであっても良い。不揮発性メモリ１２２はセキュア鍵ストアを含んでもよく、一実施形態では同不揮発性メモリ１２２はオンチップメモリである。
【００１４】
図１の例では、動作において、レジストレーション(registration)用又はアクティベーション(activation)用プロトコルが、サーバ１０２によって起動される（或いは、クライアント１０６が、レジストレーション又はアクティベーションを起動するようにしても良い）。一実施形態では、プロトコルは、装置の同一性(a device identity)を登録(register)したり、証明書データベース１１４に証明書を与えたりする（certificate into）役割を果たす。そのために、乱数発生器１１６は、乱数Ｒを生成し、サーバ１０２の証明書リクエストモジュール１１０は、デバイス証明書のリクエストを生成する。乱数Ｒとデバイス証明書のリクエストは、インターフェース１１８を介して、ネットワーク１０４へと送られる。
【００１５】
乱数Ｒとデバイス証明書のリクエストは、クライアント１０６のインターフェース１２４で受け取られる。クライアント１０６の証明書作成モジュール１２０は、証明書Ｃｅｒｔを作成する。証明書Ｃｅｒｔを作成するのに使用されるアルゴリズム例としては、図７を参照して以下に説明する。証明書作成モジュール１２０は、デバイス・プライベートキーを使用し、乱数Ｒ上の署名Ｓｉｇを算出する。オペランド(operands)は、不揮発性メモ

リ１２２に格納されているが、それらは例えばセキュリティカーネル（図５参照）にあるようにしても良い。また別の実施形態では、上記演算には、装置ＩＤ、シリアルナンバー、リージョンコード、或いはその他幾つかの数値を含む場合もある。クライアント１０６のインターフェース１２４は、ネットワーク１０４に対し、乱数Ｒや何らかの任意データ、証明書Ｃｅｒｔ、及び証明書Ｓｉｇを返す。

【００１６】

サーバ１０２Ｒは、インターフェース１１８で、乱数Ｒ、任意データ、証明書Ｃｅｒｔ、及び署名Ｓｉｇを受けとる。サーバ１０２における証明書検証モジュール１１２は、信頼できる証明書チェーン(trusted certificate chain)を用いて、証明書Ｃｅｒｔを認証し、証明書Ｃｅｒｔを用いて署名Ｓｉｇを認証し、更に乱数Ｒが、元々サーバ１０２によってクライアント１０６に送られた乱数Ｒと同じであるか否かを検証する。これらの認証及び検証が成功裏に完了したならば、サーバ１０２は、証明書データベース１１６に証明書Ｃｅｒｔを取り込む。この時点で、クライアント１０６が、サーバ１０２から（或いはクライアント１０６に許可を与える証明書を使用できる、その他幾つかのロケーションから）権利管理コンテンツ(rights managed content)やその他のオペレーションのためのデジタルライセンスを獲得することについて、推定上許可されることになる。

【００１７】

その他の実施形態としては、装置がＲＮＧを使って新しい鍵のペア {pvt1、pub1} を作成し、署名者としてプライベートキーがプログラミングされた装置を使用して、この新しい公開鍵pub1のために証明書が作成される場合もある。この新しい鍵pvt1は、乱数Ｒを持つメッセージに署名するのにつかわれる場合もある。

【００１８】

尚、ＳＳＬなどのセキュアネットワーク・プロトコルや短期秘密鍵を必要とする他のサービスなどは通常、一連の乱数群の情報源(source)を利用している。それに限定されるものではないが、図６を参照して以下に説明するようなセキュア製造プロセスは、装置に秘密乱数Ｓをシード(seed)するために使用されることができる。ＡＥＳやＳＨＡの機能のように暗号プリミティブを使った疑似乱数発生アルゴリズムは、疑似乱数(PRNs)を生成するために使用できる。このシーケンスは、装置の電源スイッチを切ってすぐに入れ直した後に、繰り返して行うべきではない。チップ型不揮発性メモリ(chip non-volatile memory)を含む状態保存メカニズム(state-saving mechanism)の使用は、高レベルのセキュリティを確実にする。シーケンス番号を格納するために、この装置は、書き換え可能な不揮発性メモリの一部を使用する。

【００１９】

図２は、システム１００に適した装置の電源オン(power up)、及び電源オフ(power down)方法の一例のフローチャート２００を示している。図２の例において、フローチャート２００は、モジュール２０２でスタートし、ここで装置の電源がオン状態にされる。図２の例で、フローチャート２００は、モジュール２０４に進み、ここでランタイム状態が１に初期化される。ランタイム状態が時間と共にインクリメントされるため、このランタイム状態は、オンチップ書込み可能メモリのような書き込み可能メモリに格納されるべきである。

【００２０】

図２の例で、フローチャート２００は、モジュール２０６に進み、装置は該シーケンス番号をインクリメントし、鍵＝ｆｎ（Ｓ、シーケンス番号）（但し、Ｓはプログラミングされた秘密シード乱数）を算出する。乱数Ｓはプログラミングされたものであるため、これをオンチップの不揮発性リードオンリメモリ（ＲＯＭ）に格納することができる。この時点で、この装置は“稼働状態(up and running)”であると見なされる。

【００２１】

図２の例で、フローチャート２００は、モジュール２０８に進み、ここでは乱数のリクエストに応えて、装置は、乱数＝ｆｎ（鍵、状態）を作成すると共に、状態を状態＋＋にインクリメントする。図２の例で、フローチャート２００は、次に決定ポイント２１０に

進み、ここでは別の乱数のリクエストを受けているか否かが判定される。別の乱数のリクエストを受け取ったと判定されたならば（２１０－Ｙ）、フローチャート２００はモジュール２０８に戻る。このようにして、モジュール２０８は、複数の乱数のリクエストのために、何度も繰り返して行われるかもしれない。

【００２２】

別の乱数のリクエストがないと判定されたならば（２１０－Ｎ）、フローチャート２００はモジュール２１２に進み、装置の電源がオフ状態にされ、それまでの装置状態は失われる。即ち、フローチャート２００は、パワーオンからパワーオフまでの間における装置の状態を示している。装置の電源が再びオン状態にされた場合、その時には新しい鍵を演算しなければならず、装置状態も再び初期化されることになる。

【００２３】

図３は、一回だけデバイス証明書を作成する方法の一例のフローチャート３００を示している。図３の例では、フローチャート３００は、モジュール３０２で始まり、ここでデバイス証明書がセキュアデバイスで作成される。次いでフローチャート３００は、モジュール３０４に進み、該デバイス証明書がシステムの外部記憶装置に格納される。装置はセキュア（技術的に安全性が保証されたもの）であるために、この変化(variation)は注目に値するが、デバイス証明書は公知のものである。従って、それはその都度、再生されないが、証明書は依然としてセキュアなものである。

【００２４】

図４は、図１乃至図３を参照して上述した技術の実施に適するコンピュータ・システム４００を示している。コンピュータ・システム４００は、コンピュータ４０２、I/0デバイス４０４、及びディスプレイデバイス４０６を含む。コンピュータ４０２は、プロセッサ４０８、通信インターフェース４１０、メモリ４１２、ディスプレイコントローラ４１４、不揮発性記憶装置４１６、及びI/0コントローラ４１８を含む。コンピュータ４０２は、I/0デバイス４０４とディスプレイデバイス４０６に接続する形でも、あるいはそれらを含む形でも良い。

【００２５】

コンピュータ４０２は、モデムやネットワークインターフェースを含んでもよい通信インターフェース４１０を介し、外部システムと接続する。通信インターフェース４１０は、コンピュータ・システム４００の一部かコンピュータ４０２の一部として考えることができる。通信インターフェース４１０は、アナログ・モデム、ISDNモデム、ケーブルモデム、トークン・リング・インタフェース、衛星通信インターフェース（例えば、“ダイレクトＰＣ”）、又は１つのコンピュータ・システムを他のコンピュータ・システムに接続する他のインターフェースのいずれでも良い。従来のコンピュータは通常、一種の通信インターフェースを備えるが、インターフェースを含まないコンピュータを作成することで、通信インターフェース４１０を厳密な意味で任意的なものとすることができる。

【００２６】

プロセッサ４０８は、それに限定されない一例として、インテル社のペンティアム（登録商標）マイクロプロセッサやモトローラ社のパワーＰＣマイクロプロセッサなどの従来マイクロプロセッサを具備しても良い。プロセッサ４０８は、総ての従来型コンピュータにとって重要な部品と言えるが、ここで説明した技術を実施するためには、適用可能な如何なる公知の（又は手頃な）プロセッサを使用することも可能である。メモリ４１２は、バス４２０によって、プロセッサ４０８に接続される。メモリ４１２（以下、“主記憶装置(primary memory)”と呼ぶこともある）は、ダイナミック・ランダムアクセス・メモリ（ＤＲＡＭ）を備えたり、またスタティック・ラム（ＳＲＡＭ）を備えたりすることも可能である。バス２２０は、プロセッサ４０８をメモリ４１２に接続したり、又不揮発性記憶装置４１６にも、ディスプレイコントローラ４１４にも、更にI/0コントローラ４１８にも接続したりすることができる。

【００２７】

I/0デバイス４０４は、キーボード、ディスクドライブ、プリンタ、スキャナ、及びマ

ウスか他のポインティング・デバイスを含む他の入出力装置を含むことができる。説明のため、これらI/0デバイスの少なくとも１つを、DVDプレーヤーのようなブロックベースドメディアデバイスと仮定する。ディスプレイコントローラ４１４は、公知の又は手頃な方法で、ディスプレイデバイス４０６上のディスプレイを制御しても良く、ディスプレイデバイス４０６は、例えば、ブラウン管（ＣＲＴ）だったり、液晶ディスプレイ（ＬＣＤ）だったりする。

【００２８】

ディスプレイコントローラ４１４とI/0コントローラ４１８はデバイスドライバを具備しても良い。デバイスドライバは、ハードウェアデバイスとの相互作用を可能にするべく開発された特定のコンピュータ・ソフトウェアである。通常、これは、デバイスと通信するためのインターフェースを構成し、ハードウェアが接続されるバスや通信サブシステムを介し、デバイスへの命令送信及び／又はデバイスからのデータ受信を実行する一方、必要不可欠なものはOSとソフトウェアアプリケーションと相互作用する。

【００２９】

デバイスドライバは、OS特有でもあるハードウェア依存型コンピュータ・プログラムを含んでもよい。そのコンピュータ・プログラムにより、別のプログラム（通常OS、アプリケーションソフトウェアパッケージ、又はOSカーネル上で動くコンピュータ・プログラム）は、透過的にハードウェアデバイスと相互作用することができ、同コンピュータ・プログラムは通常、ニーズと調和した如何なる必要な非同期・時間依存型ハードウェアにも必要不可欠な割り込み処理を提供する。

【００３０】

往々にして不揮発性記憶装置４１６（以下、“補助記憶装置(secondary memory)”と呼ぶこともある）は、磁気ハードディスクだったり、光ディスクだったり、或いは大量データ用の他の種類の記憶装置だったりする。このデータのいくつかはしばしば、コンピュータ４０２でソフトウェアを実行している間に、ダイレクトメモリアクセスプロセスによってメモリ４１２に書き込まれる。不揮発性記憶装置４１６は、ブロックベースドメディアデバイスを含むものでも良い。“機械可読メディア”や“コンピュータ可読メディア”という用語は、プロセッサ４０８によってアクセス可能であって、データ信号を符号化する搬送波を包含する如何なる公知の（又は手頃な）記憶装置をも含んでいる。

【００３１】

コンピュータ・システム４００は、異なったアーキテクチャを持っている多くの実行可能なコンピュータ・システムの一例である。例えば、インテルマイクロプロセッサをベースとするパーソナルコンピュータは往々にして複数のバスを有し、その１つは周辺機器へのI/0バスであったり、プロセッサ４０８とメモリ４１２を直接接続するもの（しばしば“メモリバス”と呼ばれる）であったりする。これらのバスは、バスプロトコルが異なることによって必要とされる如何なる変換(translation)を実行できるブリッジコンポーネントを介して接続される。

【００３２】

ネットワーク・コンピュータは、ここで提供した教示に関連して使用可能な別のタイプのコンピュータ・システムである。通常、ネットワーク・コンピュータはハードディスクや他の大容量記憶装置を備えておらず、実行可能プログラムは、プロセッサ４０８による実行のために、ネットワーク接続からメモリ４１２に取り込まれる。当該技術では既知のウェブテレビシステムも又、１つのコンピュータ・システムと考えることもできるが、ある入力や出力装置のように図４に示した特徴の幾つかを欠く可能性がある。通常、一般的なコンピュータ・システムは、少なくともプロセッサ、メモリ、メモリをプロセッサに接続するバスを具備することになる。

【００３３】

オペレーティングシステム（OS）でコンピュータ・システム４００を制御しても良い。OSは、全部ではないが殆どのコンピュータ・システムに使用されるソフトウェアプログラムであって、コンピュータのハードウェア及びソフトウェア・リソースを管理している。

通常、OSは、メモリを制御して割り当てをしたり、システムの要求に優先順位を付けたり、入出力装置を制御したり、ネットワーキングを補助したり、ファイルを管理するといったような基本タスクを実行するものである。パーソナルコンピュータ用オペレーティングシステムの例としては、マイクロソフト社のウインドウズ（登録商標）、リナックス（登録商標）、マックOS（登録商標）などがある。OSとアプリケーション・ソフトウエアとの間の線引きは往々にしてかなり難しい。幸いにも、何らかのリーズナブルな線引きで十分であるため、本願で説明した技術を理解するのにこの線引きは不要である。

【００３４】

最も低いレベルのOSとしては、その（OSの）カーネルであっても良い。通常、カーネルは、システムブートや、スタートアップする際にメモリに取り込まれるソフトウェアの第１の層(first layer)である。カーネルは、他のシステムやアプリケーション・プログラムに対する様々な共通コアサービスへのアクセスを提供する。

【００３５】

ここに使用されたように、コンピュータメモリ中のデータ・ビットにおける操作のアルゴリズム的記述と象徴(symbolic representations)は、当業者に対し最も効果的に技術を伝えるものと思われる。アルゴリズムという用語は、ここで使用するとき、また一般に使用されるとき、所望の結果を導き出す動作(operations)の首尾一貫したシーケンス（self-consistent sequence）と考えられる。この動作は、物理量の物理的な操作(manipulations)を必要とするものである。必ずしも必要ではないが、通常、こうした量は、格納、転送、結合、比較、及び他の操作を行うことができる電気、または磁気の信号の形を取る。時には、主に一般的な使用のために、こうした信号をビット(bits)、値(values)、要素(elements)、記号(symbols)、文字(characters)、項(terms)、数字(numbers)などと呼ぶのが便利であることがわかっている。

【００３６】

しかしながら、これら、及びこれらと同様の用語はすべて、適切な物理量に関連付けられ、単にこうした量に適用される手頃なラベル(labels)にすぎないことを心に留めておくべきである。特に明記しない限り、または下記の記述から明らかであるように、“処理する(processing)”、“演算する(computing)”、“計算する(calculating)”、“決定する(determining)”、“表示する(displaying)”などの用語は、コンピュータ・システムのレジスタおよびメモリ内の物理的な（電子）量として表されるデータを操作し、コンピュータ・システムのメモリまたはレジスタ、または他のこうした情報記憶装置、送信装置または表示装置内の同じように物理（電子）量として表される他のデータに変換するコンピュータの動作および処理(process)を指す。

【００３７】

ここで説明した技術を実施するための装置は、要求された目的のためにそれ専用に構成しても良く、又コンピュータに格納されたコンピュータ・プログラムによって選択的に起動(activated)されたり再構成(reconfigured)されたりする汎用コンピュータから成るものでも良い。そのようなコンピュータ・プログラムは、それらに限定されるものではないが例えば、リードオンリメモリ(ROM)、ランダムアクセスメモリ(RAM)、EPROM、EEPROM、磁気又は光学カード、フレキシブルディスクを含む如何なるタイプのディスク、光ディスク、CD-ROM、DVD、光磁気ディスク、或いは電子的な指示(instructions)を格納するのに適した如何なる公知の（又は手頃な）の媒体のような、コンピュータ可読記憶媒体に格納されても良い。

【００３８】

ここに提示されたアルゴリズムとディスプレイは本来、如何なる特定コンピュータ・アーキテクチャにも関連するものではない。この技術は、高いレベル（例えば、C/C++）であろうと低レベル（例えば、アセンブリ言語）だろうと、又インタープリタ型（例えば、Perl）だろうと、コンパイラ型（例えば、C/C++）だろうと、バイトコード（例えば、Java（登録商標））からコンパイルされるジャストインタイム（JIT）であろうと、如何なる公知の（又は手頃な）プログラミング言語を使用して実施されても良い。どんな公知の（

又は手頃な）コンピュータでも、アーキテクチャに関係なく、コンパイルされた機械コードか、さもなければ何らかの言語からコンピュータ・アーキテクチャと互換性がある機械コードへとアセンブルされた機械コードを実行することが可能でなければならない。

【００３９】

図５は、図１乃至図３を参照して上述した技術を実施するのに適していたセキュアシステム５００の一例を示している。代表的なセキュアシステム５００は、ゲーム機、メディアプレーヤー、埋め込み型(embedded)セキュアデバイス、セキュアプロセッサ付き“従来型”PC、或いはセキュアプロセッサを備える他のコンピュータ・システムを備えても良い。

【００４０】

図５の例においてセキュアシステム５００は、セキュアプロセッサ５０２、OS５０４、チケットサービス５０６、呼出しアプリケーション(calling application)５０８、及びプロテクト・メモリ５１０を含む。図５の例ではOS５０４は、鍵ストア５１６、暗号化/復号化エンジン５１７、及びセキュリティAPI５１８を順に含むセキュリティカーネル５１４を有する。上述したこれら部品の１つ又はそれ以上、或いはその一部を、プロテクト・メモリ５１０又は保護されていないメモリ（図示せず）に属させるようにしても良いことを留意すべきである。

【００４１】

さらに留意すべきは、慣例だけによって、セキュリティカーネル５１４が、OS５０４に属するように表現されていることである。実際にOS５０４の一部であってもなくても良く、OSの外側に存在することも、或いはOSを含まないシステム上に存在することも可能である。説明を簡単にする目的で、OS５０４は認証可能であることが前提である。実施形態では、チケットサービス５０６は、OS５０４の一部であっても良い。認証付きのチケットサービス５０６を読み込むこと(loading)が、セキュリティを向上できるという理由で、これは望ましいかもしれない。従って、そのような実施形態では、OS５０４は認証付きで読み込まれ、チケットサービス５０６を含むことになる。

【００４２】

説明の簡略化のために、プロテクト・メモリは単一メモリとして示される。しかしながら、このプロテクト・メモリは、保護された主記憶装置、保護された補助記憶装置、及び／又は、シークレットメモリを含んでも良い。メモリが保護されるのを確実とするためには、公知の（又は手頃な）メカニズムが適所にあることが前提である。主記憶装置と補助記憶装置との間、及び／又は、揮発性記憶装置と不揮発性記憶装置との間のインタープレイ(interplay)は、公知であるため、様々な種類のメモリと記憶装置の間の区別は図５に関しては示されていない。

【００４３】

チケットサービス５０６は、例えば“デジタルライセンス有効化サービス”として考えられても良く、また非限定的な実施形態では、ライセンス有効化に関連する公知の（又は手頃な）手順を含んでもよい。例えば、チケットサービス５０６は、デジタルライセンスを有効にするための手順、PKI有効化手順などを含んでも良い。図５の例では、チケットサービス５０６は、呼び出しアプリケーション５０８からチケットを有効化することができる。この動作において、チケットサービス５０６は、呼出しアプリケーション５０８からチケットを獲得し、続けてチケット有効化を進行させる。

【００４４】

チケットには、自分の名前を記入することができる(personalized)。この場合には、秘密共有暗号鍵を算出するために、（上述したようにプログラミングされた）デバイス・プライベートキーを用いて復号化されることができる。チケットは、インターネット・ダウンロードメカニズムを使って獲得されてもされなくてもよく、また、書き込み可能なフラッシュメモリに記憶されてもされなくてもよい。

【００４５】

一実施形態では、セキュリティカーネル５１４が、起動時点において読み込み(loaded)

されてもよい。別の実施形態では、セキュリティカーネルの一部が、起動時点において読み込みされ、残りが後で読み込みされても良い。この技術の一例は、Srinivasanらによって、2003年2月7日に出願された“信頼性や下位互換性のあるプロセッサ、及びその上でのセキュアソフトウェア実行”というタイトルの出願番号：１０／３６０，８２７号に記載される。セキュアな方法（技術的に安全性が保証された方法、in a secure manner）で、セキュリティカーネル５１４を読み込みするべく、如何なる公知の（又は手頃な）技術も使用可能である。

【００４６】

鍵ストア５１６は、鍵のための格納場所群からなる。鍵を格納するのに使用されるデータ構造は重要な意味をもつものではないが、鍵ストア５１６は、鍵の一配列と思われても良い。鍵を格納するために、適用可能な如何なる公知の（又は、手頃な）構造も使用可能である。非限定的な実施形態では、鍵ストア５１６は、静的な(static)鍵を以て初期化されるが、可変鍵(variable keys)は初期化されない（或いは、セキュアでない値に初期化される）。例えば、幾つかの鍵ストア位置(key store locations)は、認証されたセキュリティカーネル５１４の読み込み(loading)の一部としての信頼値（例えば信頼ルート鍵）で予備充填(pre-filled)される。不揮発性メモリにおけるプライベートキーは、ここから取り出すことも、また将来の使用に備えて鍵ストアに格納することも可能である。

【００４７】

一実施形態において、暗号化/復号化エンジン５１７は、暗号化と復号化の両方が可能である。例えば、かかる動作において、アプリケーションは、セキュリティAPI５１８に、アプリケーションが暗号化に使用できるというキーハンドルを要求するかもしれない。暗号化/復号化エンジン５１７を、キーハンドルを使用してデータを暗号化するのに使用しても良い。好都合なことには、セキュリティAPI５１８は、キーハンドルを平文で提供するが、鍵そのものは決してセキュリティカーネル５１４から離れることはない。

【００４８】

セキュリティAPI５１８は、鍵を明らかにすることなく（すなわち、鍵がセキュリティカーネル５１４を離れないか、或いは暗号化される場合にだけに限って鍵がセキュリティカーネル５１４を離れるような状態で）、鍵ストア５１６内の鍵を使用した動作を実行することができる。セキュリティAPI５１８は、鍵ストア５１６で鍵（そして潜在的には、他のセキュリティマテリアル）を作成(create)し、投入(populate)し、使用するためのサービスを含んでも良い。また、一実施形態では、セキュリティAPI５１８は、また、秘密鍵とデバイス・プライベートキーを含む、内部の秘密と不揮発性データへのアクセスを与える。例えば、デバイス・プライベートキーは、鍵ストアに格納されたり、セキュリティAPIに使用されたりするかもしれない。１つのAPI呼び出し(API call)が、（ここで記述した、証明書を作成するためのアルゴリズムを用いて）デバイス証明書を戻すために使用されることも可能である。また復号化のための共用鍵を算出するためにプライベートキーを使用したり、メッセージや証明書に署名するためにプライベートキーを使用したりするために、別のAPI呼び出しが構成されることが可能である。実装の形態によっては、セキュリティAPI５１８は、ハードウェア加速を使用することでAES及びSHAオペレーションをサポートするようにしても良い。

【００４９】

図５の例では、チケットサービス５０６とセキュリティAPI５１８は、システムセキュリティのための個々の実行スペースにおいて実行してもよい。データブロックを有効化するために、チケットサービス５０６は、ヘッダのデータを使ってチケットを有効にしても良い。そのチケットは、暗号化された鍵を含んでも良い。チケットサービス５０６は、セキュリティカーネル５１４におけるサービス（例えば、暗号化/復号化エンジン５１７）を利用して、鍵を復号化する。

【００５０】

一実施形態では、暗号化/復号化エンジン５１７は、この復号化を実行するために、鍵ストア５１８からの秘密共通鍵を使用する。別の実施形態では、チケットサービス５０６

は、フラッシュかネットワーク（図示せず）から得られたデバイス個別チケット(device personalized ticket)を使用して、コンテンツに対する幾つかの権利を有効にして、その後に鍵を戻すことも可能である。いずれにしても、このプロセスは、鍵を戻す。この個別チケットは、デバイス・プライベートキーの関数(function)であって、かつ不揮発性メモリにプログラミングされた鍵により暗号化されることができる。

【００５１】

矢印５２０～５２８を以て、システム５００のデータフローの例が例示を目的として供される。チケットサービス５０６での証明書リクエストの受取りは、呼出しアプリケーション５０８からチケットサービス５０６への証明書リクエスト矢印５２０によって表わされる。

【００５２】

チケットサービス５０６からセキュリティAPI５１６への証明書リクエストの送りは、証明書リクエスト矢印５２２によって表されている。セキュリティカーネル５１４において、公開鍵／デバイス証明書・作成エンジン５１７は、鍵/署名ストア５１８からの鍵／署名データにアクセスする。このアクセスは、プライベートキー／署名アクセス矢印５２４によって表されている。セキュリティAPI５１６は、デバイス証明書矢印５２６に示すように、デバイス証明書をチケットサービス５０６に返し、それは更にデバイス証明書矢印５２８に示すように、呼出しアプリケーション５０８に転送される。

【００５３】

図６は、セキュアデバイスを製造する方法の例を示すフローチャート６００である。この方法、及び他の方法は、連続的に並べられたモジュールの形で表現される。しかしながら、これらの方法における各モジュールは、並べ替えされたり、並列実行のために適当に並べられたりしても良い。図６の例において、フローチャート６００は、モジュール６０２でスタートし、ここで装置ＩＤが取得される。装置ＩＤは、シリアルナンバーであっても、或いは装置のための他の何らかの固有な識別子であっても良い。

【００５４】

図６の例で、フローチャート６００は、モジュール６０４に進み、ここでは装置のために小署名プライベートキー(a small-signature private key)として使用する目的で、擬似乱数が供される。今日に至るまで、本当の意味での乱数というものはコンピュータ上では生成できないが、当然のことながら、疑似乱数発生器や外部のセキュアなハードウェアにおける真性乱数発生器が、ここで意図する目的のために使用できるだろう。小署名プライベートキーは、これに限定されない例として、楕円カーブプライベートキーや比較的小さなフットプリント(footprint)を持った他の何らかのプライベートキーでも良い。

【００５５】

図６の例で、フローチャート６００は、モジュール６０６に進み、ここでは共通パラメータを使用してプライベートキーから公開鍵が算出される。例えば、スカラー倍数(scalar multiple)がプライベートキーとなるように、基点(base point)の倍数が算出されても良い。

【００５６】

図６の例で、フローチャート６００は、モジュール６０８に進み、ここでは一定の証明書構造(fixed certificate structure)が、証明書を作成するために使用される。証明書は、例えば楕円カーブＤＳＡなどの小署名アルゴリズム(small signature algorithm)を使用して署名される。一実施形態では、該一定の証明書構造は、少なくとも装置ＩＤ、発行者名、及びデバイス公開鍵を含むかもしれない。小署名アルゴリズムは、署名サイズを最小限におさえるのに使用される。これに限定されない例として、楕円カーブ署名アルゴリズムが使用されても良い。

【００５７】

図６の例で、フローチャート６００は、モジュール６１０に進み、ここでは｛装置ＩＤ、プライベートキー、発行者ＩＤ、署名｝が、装置の不非揮発性メモリにプログラミングされる。このセットは、これらのアイテムが殆どの目的に対して充分セキュリティを提供

するという理由でこれら４つのアイテムを含んでおり、また該セットは、比較的小さいサイズのプライベートキーと署名のおかげで比較的小さいフットプリントを有する（推定するに、装置ＩＤ及び発行者ＩＤも又、比較的小さいフットプリントを有している）。一実施形態では、例えば公開鍵など、デバイス証明書の作成(construct)に必要とされる如何なるデータも、要求に応じてプログラミングされた状態で生成されても良い。しかしながら、与えられた実施形態や実装に見合った形で、より多くのアイテム、又より少ないアイテムが不揮発性メモリにプログラミングされ得る。

【００５８】

図６の例では、フローチャート６００は、モジュール６１２に進み、秘密の乱数が装置のＲＯＭにプログラミングされる。この秘密乱数は、疑似乱数的に生成されても、また任意に割り当てられるようにしても良い。この秘密乱数は、セキュアな擬似乱数の生成をサポートするのに利用できる。また別の実施形態では、ＲＯＭが、他の公知の（又は手頃な）不揮発性記憶装置と置き換えられても良い。

【００５９】

図７は、セキュア証明書の作成方法の例を示したフローチャート７００である。メリットとして、この方法は、不揮発性でプログラミングされた鍵と必要なソフトウェアを有する装置が、装置認証に使用可能な完全なデバイス証明書を作成することを可能にする。図７の例では、フローチャート７００は、モジュール７０２でスタートし、ここではデバイス証明書のリクエストが、呼出しアプリケーションから受け取られる。

【００６０】

図７の例で、フローチャート７００は、モジュール７０４に進み、ここでは｛装置ＩＤ、プライベートキー、発行者ＩＤ、署名｝が、不非揮発性メモリから読み込まれる。一実施形態では、セキュリティカーネル・モジュールが、不揮発性メモリにアクセスして読み込む。この目的のために適切なセキュリティカーネル・モジュールの例としては、Srinivasanらによって２００３年２月７日に出願された“信頼性や下位互換性のあるプロセッサ及びその上でのセキュアソフトウェア実行”というタイトルの米国出願第１０／３６０，８２７号や、Srinivasanらによって２００６年１０月２４日に出願された“セキュアデバイス認証システム及び方法”というタイトルの米国出願第１１／５８６，４４６号に記載されている。しかしながら、如何なる適用可能な公知の（又は手頃な）セキュリティカーネル・モジュールでも使用可能である。

【００６１】

図７の例で、フローチャート７００は、モジュール７０６に進み、ここでは公開鍵が、プライベートキーと（もしあるならば）共通パラメータとによって算出される。一実施形態では、この演算は、図６を参照して上述した方法のような製造プロセスに使用されたのと同じアルゴリズムを利用する。公開鍵は、セキュリティカーネルで算出されても良い。

【００６２】

図７の例で、フローチャート７００は、モジュール７０８に進み、ここではデバイス証明書が、装置ＩＤ、発行者ＩＤ、公開鍵、署名、及び共通パラメータから作成される。一実施形態において、セキュリティカーネル・モジュールは、図６を参照して上述した方法のような製造プロセスで使用されおり、該デバイス証明書の構造を認識している。メリットとして、デバイス証明書は、要求に応じて作成できる。

【００６３】

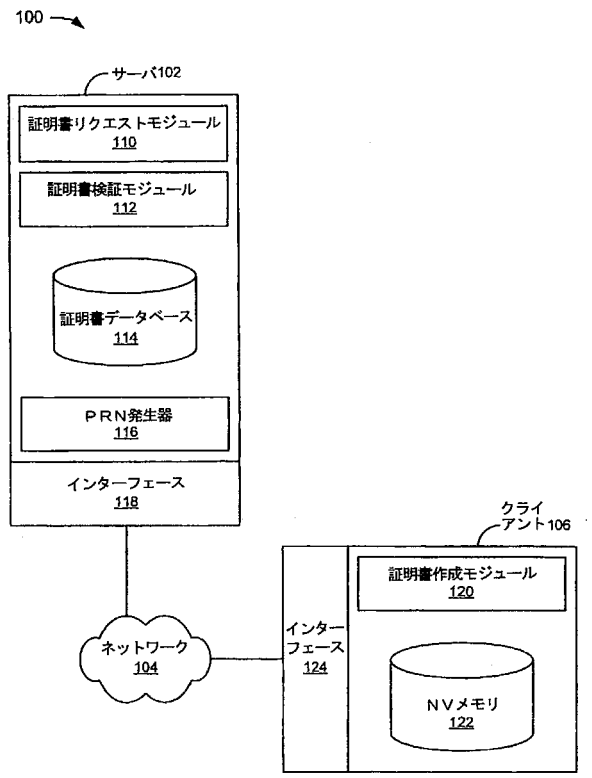
図７の例で、フローチャート７００は、モジュール７１０に進み、ここではデバイス証明書が呼出しアプリケーションに供される。デバイス証明書が呼出しアプリケーションに供された時、フローチャート７００は終了する。この方法は、別の呼出しアプリケーションで（何らかの理由で、デバイス証明書が再び必要とされる時は同じ呼出しアプリケーションによって）再開することが可能である

【００６４】

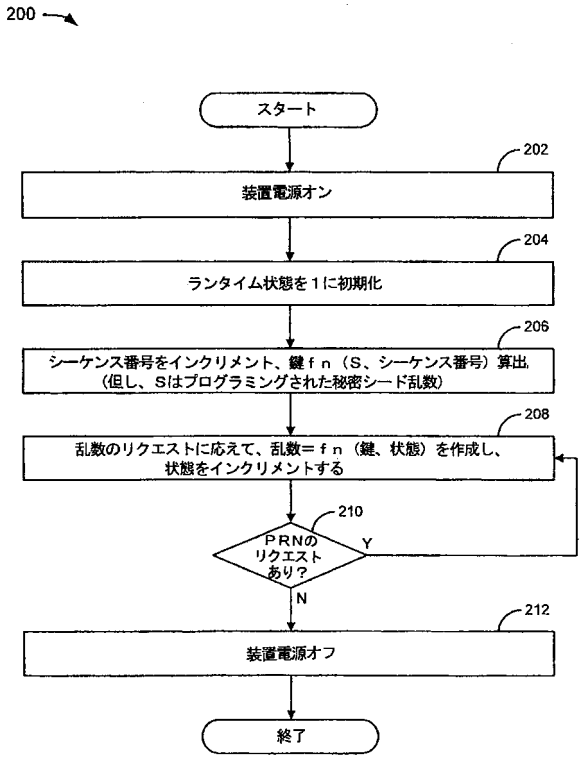
ここで使用されたように、用語“コンテンツ”は、メモリに保存可能な如何なるデータをも広く含むことを意図する。

【００６５】
ここで使用されたように、用語“実施形態”は、それに限定されない一例として説明を行うための実施例を意味している。
【００６６】
当業者であれば前述した実施例と実施形態は、代表的なものであり、本発明の範囲を限定するものでないことが理解されるであろう。当業者が明細書を考慮し、図面を検討したときに自明な置き換え、増強、均等物やそれらへの改良が、本発明の要旨と範囲に含まれることを意図している。したがって、添付された特許請求の範囲は、本発明の要旨と範囲内にあるこのような総ての変更、置き換え、均等物を含むことを意図している。
【符号の説明】
【００６７】
１００ システム
１０２ サーバ
１０４ ネットワーク
１０６ クライアント
１１０ 証明書リクエストモジュール
１１２ 証明書検証モジュール
１１４ 証明書データベース
１１６ 擬似乱数（ＰＲＮ）発生器
１１８ インターフェース
１２０ 証明書作成モジュール
１２２ 不揮発性（ＮＶ）メモリ
１２４ インターフェース
２００ フローチャート
２０２、２０４、２０６、２０８、２１２ モジュール
２１０ 決定ポイント
３００ フローチャート
３０２、３０４ モジュール
４００ コンピュータ・システム
４０２ コンピュータ
４０４ I/Oデバイス
４０６ ディスプレイデバイス
４０８ プロセッサ
４１０ 通信インターフェース
４１２ メモリ
４１４ ディスプレイコントローラ
４１６ 不揮発性記憶装置
４１８ I/Oコントローラ
４２０ バス
５００ セキュアシステム
５０２ セキュアプロセッサ
５０４ OS（オペレーティングシステム）
５０６ チケットサービス
５０８ 呼出しアプリケーション
５１０ プロテクト・メモリ
５１４ セキュリティカーネル
５１６ 鍵ストア
５１７ 暗号化/復号化エンジン
５１８ セキュリティAPI

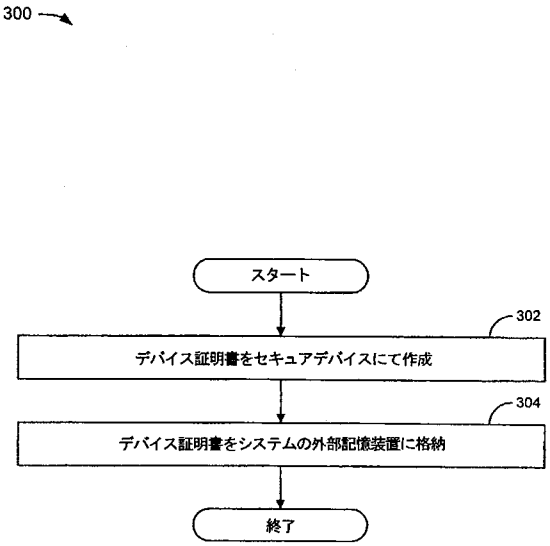
【図１】


100
サーバ102
証明書リクエストモジュール
110
証明書検証モジュール
112
証明書データベース
114
PRN発生器
116
インターフェース
118
ネットワーク
104
インターフェース
124
クライアント106
証明書作成モジュール
120
NVメモリ
122

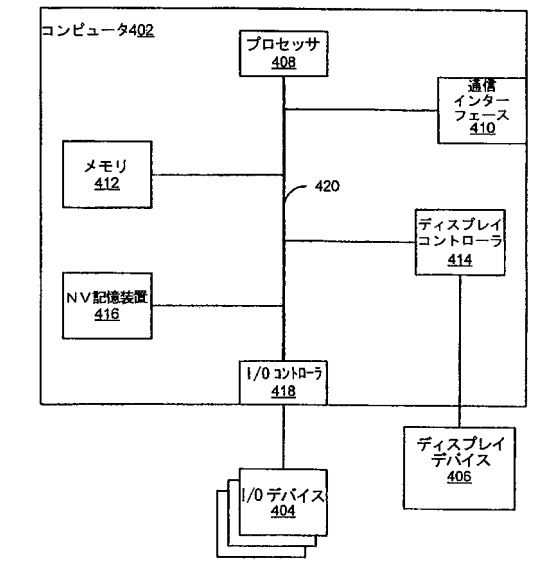
【図２】


200
スタート
202
装置電源オン
204
ランタイム状態を１に初期化
206
シーケンス番号をインクリメント、鍵ｆｎ（Ｓ、シーケンス番号）算出
（但し、Ｓはプログラミングされた秘密シード乱数）
208
乱数のリクエストに応えて、乱数＝ｆｎ（鍵、状態）を作成し、
状態をインクリメントする
210
PRNの
リクエスト
あり？
Y
N
212
装置電源オフ
終了

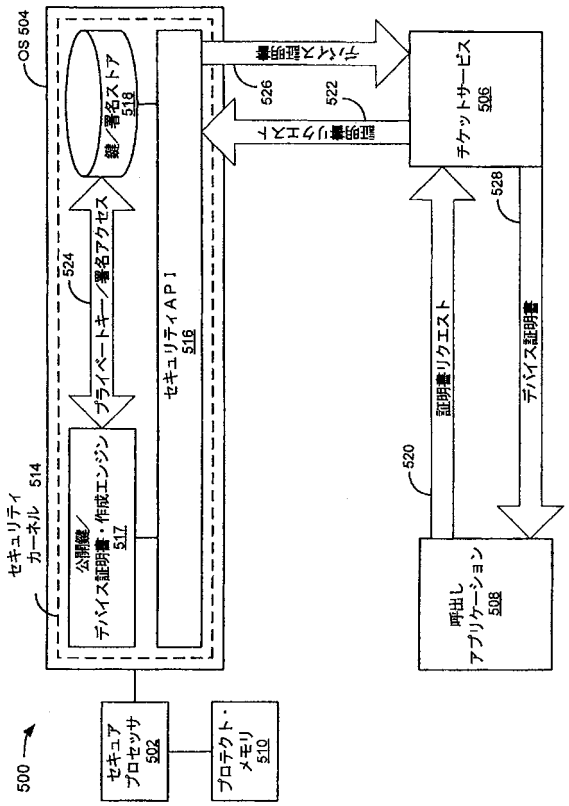
【図３】


300
スタート
302
デバイス証明書をセキュアデバイスにて作成
304
デバイス証明書をシステムの外部記憶装置に格納
終了

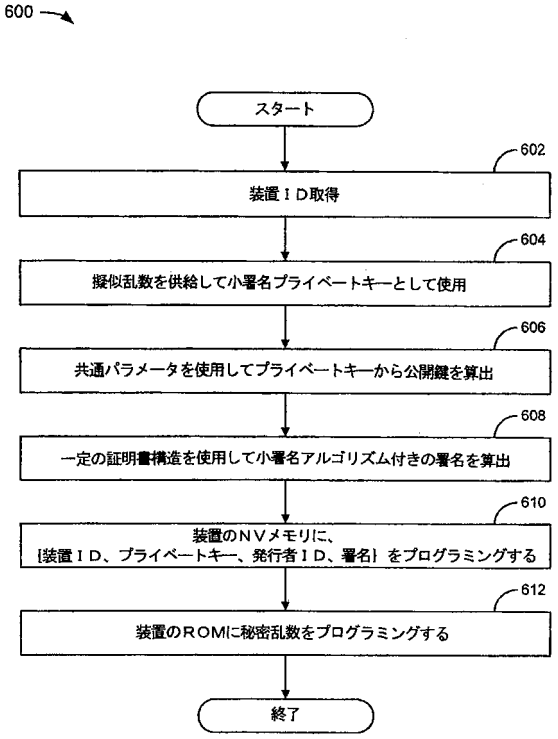
【図４】


400
コンピュータ402
プロセッサ
408
通信
インター
フェース
410
メモリ
412
420
ディスプレイ
コントローラ
414
NV記憶装置
416
I/O コントローラ
418
ディスプレイ
デバイス
406
I/O デバイス
404

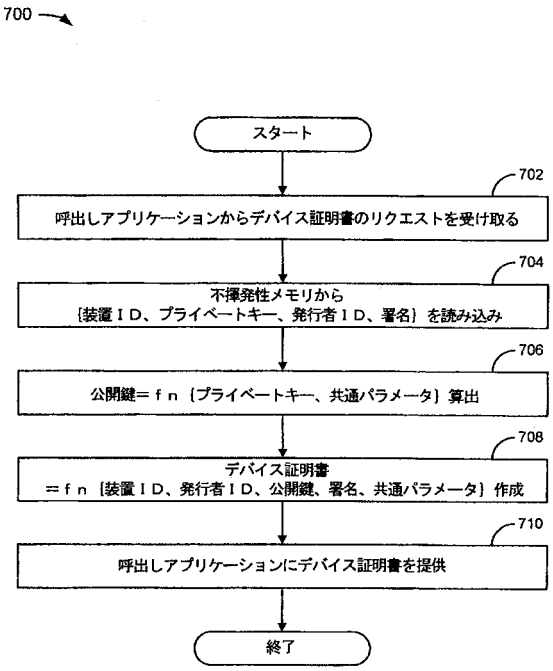
【図５】


OS 504
鍵／署名ストア 518
セキュリティAPI 516
セキュリティカーネル 514
公開鍵／デバイス証明書・作成エンジン 517
プライベートキー／署名アクセス 524
デバイス証明書 526
証明書リクエスト 522
チケットサービス 506
証明書リクエスト 520
デバイス証明書 528
呼出しアプリケーション 508
セキュアプロセッサ 502
プロテクト・メモリ 510
500

【図６】


600
スタート
602
装置ＩＤ取得
604
擬似乱数を供給して小署名プライベートキーとして使用
606
共通パラメータを使用してプライベートキーから公開鍵を算出
608
一定の証明書構造を使用して小署名アルゴリズム付きの署名を算出
610
装置のＮＶメモリに、
[装置ＩＤ、プライベートキー、発行者ＩＤ、署名] をプログラミングする
612
装置のＲＯＭに秘密乱数をプログラミングする
終了

【図７】


700
スタート
702
呼出しアプリケーションからデバイス証明書のリクエストを受け取る
704
不揮発性メモリから
[装置ＩＤ、プライベートキー、発行者ＩＤ、署名] を読み込み
706
公開鍵＝ｆｎ［プライベートキー、共通パラメータ］算出
708
デバイス証明書
＝ｆｎ［装置ＩＤ、発行者ＩＤ、公開鍵、署名、共通パラメータ］作成
710
呼出しアプリケーションにデバイス証明書を提供
終了

フロントページの続き

(72)発明者　プリンスン，ジョン
アメリカ合州国，カリフォルニア州　９５０１４，クパティーノ，プラム　ツリー　レーン　１０４３９

Ｆターム(参考) 5J104 AA07　KA05　KA06　KA21　NA02　NA37　NA38　PA07