(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
5 January 2006 (05.01.2006)

PCT

(10) International Publication Number
WO 2006/001916 A2

(51) International Patent Classification^7: H04L 9/32

(21) International Application Number: PCT/US2005/016559

(22) International Filing Date: 13 May 2005 (13.05.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/866,252 10 June 2004 (10.06.2004) US

(71) Applicant (*for all designated States except US*): **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, CA 95052 (US).

(72) Inventor; and
(75) Inventor/Applicant (*for US only*): **BRICKELL, Ernest** [US/US]; 3106 Northwest Luray Terrace, Portland, OR 87111 (US).

(74) Agents: **VINCENT, Lester, J.** et al.; Blakely, Sokoloff, Taylor & Zafman, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).

(81) Designated States (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: AN APPARATUS AND METHOD FOR PROVING THE DENIAL OF A DIRECT PROOF SIGNATURE

(57) **Abstract:** In some embodiments, a method and apparatus for proving the denial of a direct proof signature are described. In one embodiment, a trusted hardware device convinces a verifier that the trusted hardware device possesses cryptographic information without revealing unique, device identification information of the trusted hardware device or the cryptographic information. Once the verifier is convinced that the hardware device possesses the cryptographic information, the verifier may issue a denial of signature request to the trusted hardware device, including at least one compromised direct proof signature. In response, the trusted hardware device issues a denial of the compromised direct proof signature by proving to the verifier that a cryptographic key held by the trusted hardware device was not used to form the at least one compromised direct proof signature. Other embodiments are described and claims.

AN APPARATUS AND METHOD FOR PROVING THE DENIAL OF A DIRECT PROOF SIGNATURE

FIELD OF THE INVENTION

[0001] One or more embodiments of the invention relate generally to the field of cryptography. More particularly, one or more of the embodiments of the invention relates to a method and apparatus for proving the denial of a direct proof signature.

BACKGROUND OF THE INVENTION

[0002] For many modern communication systems, the reliability and security of exchanged information is a significant concern. To address this concern, the Trusted Computing Platform Alliance (TCPA) developed security solutions for platforms. In accordance with a TCPA specification entitled "Main Specification Version 1.1b," published on or around February 22, 2002, each personal computer (PC) is implemented with a trusted hardware device referred to as a Trusted Platform Module (TPM). Each TPM contains a unique endorsement key pair (EK), which features a public EK key (PUBEK) and a private EK key (PRIVEK). The TPM typically has a certificate for the PUBEK signed by the manufacturer.

[0003] During operation, an outside party (referred to as a "verifier") may require authentication of the TPM. This creates two opposing security concerns. First, the verifier needs to be sure that requested authentication information is really coming from a valid TPM. Second, an owner of a PC including the TPM wants to maintain as much privacy as possible. In particular, the owner of the PC wants to be able to provide authentication information to different verifiers without those verifiers being able to determine that the authentication information is coming from the same TPM.

[0004] One proposed solution to these security issues is to establish a Trusted Third Party (TTP). For instance, the TPM would create an Attestation Identify Key pair (AIK), namely a public AIK key and a private AIK key. The public AIK key could be placed in a certificate request signed with the PRIVEK, and subsequently sent to the TTP. The certificate for the PUBEK would also be sent to the TTP. Once the certificates are received, the TTP would check that the signed certificate request is valid, and if valid, the TTP would issue a certificate to the TPM.

[0005] Once a certificate is issued, the TPM would then use the public AIK and the TTP issued certificate when the TPM received a request from a verifier. Since the AIK and certificate would be unrelated to the EK, the verifier would get no information about the identity of the TPM or PC implemented with the TPM. In practice, the above-identified approach is problematic because it requires TTPs to be established. Identifying and establishing various parties that can serve as TTPs has proven to be a substantial obstacle.

[0006] Another proposed solution is set forth in a co-pending U.S. Application No. 10/306,336, filed November 27, 2002, which is also owned by the assignee of the present application. The proposed solution utilizes a direct proof method whereby the TPM could prove directly without requiring a trusted third party that an AIK has been created by a valid TPM without revealing the identity of the TPM. In that solution, each TPM has a unique private key. Unfortunately, an adversary may take a TPM and, using sophisticated means, extract the unique private key from the TPM.

[0007] In the Direct Proof method, there is a method given to be able to revoke a key that has been removed from a TPM. During the Direct Proof protocol, the TPM gets a base, h, and computes and reveals $k = h^f \bmod n$, where n is part of the public key, and f is part of the unique key held by the TPM. So if a verifier receives a value f0 that has been removed from a TPM, the verifier can check whether the Direct Proof was created using this value f0, by performing the computation $k0 = h^{f0} \bmod n$, and checking to see if k = k0. For if k = k0, then the Direct Proof was created using f0, and if k is not equal to k0, then the Direct Proof was created using some other private key.

[0008] One limitation of this method is that it requires that the verifier obtain the value of f0. It is conceivable that the adversary could have obtained the secret unique value from a TPM, and used it in a way that the verifier could not obtain the value of f0, but could know that for a particular k0, that value of f0 had been removed from the TPM. In U.S. Application No. 10/306,336, one method was presented for dealing with this problem. It required the verifier to provide the value of the base h for each TPM to use when interacting with that verifier. This has the property that it allows the verifier to be able to link all interactions with that verifier.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The various embodiments of the present invention are illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which:

[00010] FIG. 1 illustrates a system featuring a platform implemented with a Trusted Platform Module (TPM) that operates in accordance with one embodiment.

[00011] FIG. 2 illustrates a first embodiment of the platform including the TPM of FIG. 1.

[00012] FIG. 3 illustrates a second embodiment of the platform including the TMP of FIG. 1.

[00013] FIG. 4 illustrates an exemplary embodiment of a computer implemented with the TMP of FIG. 2.

[00014] FIG. 5 illustrates a flow diagram of a procedure to setup a TPM during manufacturing according to one embodiment.

[00015] FIG. 6 illustrates a flow diagram of a procedure to setup each platform manufactured according to one embodiment.

[00016] FIG. 7 is a flowchart illustrating a method for verifying that a cryptographic key stored within a trusted hardware device is uncompromised, in accordance with one embodiment.

[00017] FIG. 8 is a flowchart illustrating a method for a zero knowledge proof to show that two discrete logarithms are the same, in accordance with one embodiment.

[00018] FIG. 9 is a flowchart illustrating a method for conceptually illustrating the verification of a proof that two discrete logarithms are the same, in accordance with one embodiment.

[00019] FIG. 10 is a flowchart illustrating a method for convincing a verifier that a cryptographic key stored within a trusted hardware device is uncompromised, in accordance with one embodiment.

DETAILED DESCRIPTION

[00020] A method and apparatus for proving the denial of a direct proof signature are described. In one embodiment a trusted hardware device convinces a verifier of possessing cryptographic information without revealing unique, device identification information of the trusted hardware device or the cryptographic information. This is

accomplished without the use of a Trusted Third Party (TTP). Rather, it is accomplished by a "direct proof" methodology in which computations by the TPM involve exponentiations using a cryptographic key as an exponent. In one embodiment, the trusted hardware device proves to a verifier that a digital signature used in the direct proof ("direct proof signature") is based on an uncompromised cryptographic key.

[00021] In one embodiment, the verifier may issue a denial signature request to the trusted hardware device to prove that a cryptographic key held by the trusted hardware device was not used to form a direct proof signature suspected of being compromised (suspect direct proof signature). For one embodiment, the functionality of the TPM, which is configured to prove to a verifier that information (e.g., cryptographic key, digital signature, digital certificate, etc.) from the TPM is uncompromised, is deployed as firmware. However, it is contemplated that such functionality may be deployed as dedicated hardware or software. Instructions or code forming the firmware or software are stored on a machine-readable medium.

[00022] Herein, "machine-readable medium" may include, but is not limited to a floppy diskette, hard disk, optical disk (e.g., CD-ROMs, DVDs, mini-DVDs, etc.), magneto-optical disk, semiconductor memory such as read-only memory (ROM), random access memory (RAM), any type of programmable read-only memory (e.g., programmable read-only memory "PROM", erasable programmable read-only memories "EPROM", electrically erasable programmable read-only memories "EEPROM", or flash), magnetic or optical cards, or the like. It is contemplated that a signal itself and/or a communication link can be regarded as machine-readable medium since software may be temporarily stored as part of a downloaded signal or during propagation over the communication link.

[00023] In the following description, certain terminology is used to describe certain features of one or more embodiments of the invention. For instance, "platform" is defined as any type of communication device that is adapted to transmit and receive information. Examples of various platforms include, but are not limited or restricted to computers, personal digital assistants, cellular telephones, set-top boxes, facsimile machines, printers, modems, routers, or the like. A "communication link" is broadly defined as one or more information-carrying mediums adapted to a platform. Examples of various types of communication links include, but are not limited or restricted to electrical wire(s), optical fiber(s), cable(s), bus trace(s), or wireless signaling technology.

[00024] A "verifier" refers to any entity (e.g., person, platform, system, software, and/or device) that requests some verification of authenticity or authority from another entity. Normally, this is performed prior to disclosing or providing the requested information. A "prover" refers to any entity that has been requested to provide some proof of its authority, validity, and/or identity. A "device manufacturer," which may be used interchangeably with "certifying manufacturer," refers to any entity that manufactures or configures a platform or device (e.g., a Trusted Platform Module).

[00025] As used herein, to "prove" or "convince" a verifier that a prover has possession or knowledge of some cryptographic information (e.g., signature key, a private key, etc.) means that, based on the information and proof disclosed to the verifier, there is a high probability that the prover has the cryptographic information. To prove this to a verifier without "revealing" or "disclosing" the cryptographic information to the verifier means that, based on the information disclosed to the verifier, it would be computationally infeasible for the verifier to determine the cryptographic information. Such proofs are hereinafter referred to as direct proofs. The term "direct proof" refers to zero-knowledge proofs, as these types of proofs are commonly known in the field.

[00026] Throughout the description and illustration of the various embodiments discussed hereinafter, coefficients, variables, and other symbols (e.g., "h") are referred to by the same label or name. Therefore, where a symbol appears in different parts of an equation as well as different equations or functional description, the same symbol is being referenced.

I. GENERAL ARCHITECTURE

[00027] FIG. 1 illustrates system 100 featuring a platform implemented with a trusted hardware device (referred to as "Trusted Platform Module" or "TPM") in accordance with one embodiment. A first platform 102 (Verifier) transmits an authentication request 106 to a second platform 200 (Prover) via network 120. In response to request 106, second platform 200 provides the authentication information 108. In one embodiment, network 120 forms part of a local or wide area network, and/or a conventional network infrastructure, such as a company's Intranet, the Internet, or other like network.

[00028] Additionally, for heightened security, first platform 102 may need to verify that prover platform 200 is manufactured by either a selected device manufacturer or a selected group of device manufacturers (hereinafter referred to as "device manufacturer(s)

110"). In one embodiment, first platform 102 challenges second platform 200 to show that it has cryptographic information (e.g., a private signature key) generated by device manufacturer(s) 110. Second platform 200 replies to the challenge by providing authentication information, in the form of a reply, to convince first platform 102 that second platform 200 has cryptographic information generated by device manufacturer(s) 110, without revealing the cryptographic information or any unique, device/platform identification information.

[00029] FIG. 2 is a block diagram further illustrating platform 200 including TPM 220 to convince a verifier that platform 200 possesses uncompromised cryptographic information without disclosure of the cryptographic information or any unique device identification information. Representatively, computer system 200 comprises a processor system bus (front side bus (FSB)) 204 for communicating information between processor (CPU) 202 and chipset 210. As described herein, the term "chipset" is used in a manner to collectively describe the various devices coupled to CPU 202 to perform desired system functionality.

[00030] Representatively, graphics block 218 hard drive devices (HDD) 214 and main memory 212 may be coupled to chipset 210. In one embodiment, chipset 210 is configured to include a memory controller and/or an input/output (I/O) controller to communicate with I/O devices 216 (216-1, . . ., 216–N). In an alternate embodiment, chipset 210 is or may be configured to incorporate graphics block 218 and operate as a graphics memory controller hub (GMCH). In one embodiment, main memory 212 may include, but is not limited to, random access memory (RAM), dynamic RAM (DRAM), static RAM (SRAM), synchronous DRAM (SDRAM), double data rate (DDR) SDRAM (DDR-SDRAM), Rambus DRAM (RDRAM) or any device capable of supporting high-speed buffering of data.

[00031] FIG. 3 further illustrates Trusted Platform Module (TPM) 220 of second platform 200, in accordance with one embodiment. TPM 220 is a cryptographic device that is manufactured by device manufacturer(s) 110. In one embodiment, TPM 220 comprises processor unit 222 with a small amount of on-chip memory encapsulated within a package. In one embodiment, the encapsulated memory may be used to store cryptographic key 230 received from a certifying manufacturer. TPM 220 is configured to provide authentication information to first platform 102 that would enable it to determine that the authentication information is transmitted from a valid TPM. The authentication information used is non-unique data that would make it highly likely that the TPM's or second platform's identify can be determined, referred to herein as "unique, device identification information."

[00032] In one embodiment, TMP 220 further comprises non-volatile memory 224 (e.g., flash) to permit storage of cryptographic information such as one or more of the following: keys, hash values, signatures, certificates, etc. In one embodiment, the cryptographic information is a cryptographic key received from a certifying manufacturer. As shown below, a hash value of "X" may be represented as "Hash(X)". Of course, it is contemplated that such information may be stored within external memory 280 of platform 200 in lieu of flash memory 224. The cryptographic information may be encrypted, especially if stored outside TPM 220.

[00033] In one embodiment, TPM 220 includes authentication logic 240 to respond to an authentication request from a verifier platform. In one embodiment, authentication logic 240 convinces or proves to the verifier platform that TPM 220 has stored cryptographic information generated by a certifying device manufacturer, without revealing the cryptographic information or any unique device/platform identification information. As a result, authentication logic 240 performs the requested authentication while preserving the identity of the prover platform. Authentication logic 240 is further illustrated with reference to FIG. 4.

[00034] As illustrated, direct proof logic 250 is configured to engage in a direct proof, as described in further detail below, to convince a verifier that the prover platform contains the cryptographic information from a certifying manufacturer without revealing the cryptographic information. As described below, key logic 270 performs platform set-up of TPM 220 to receive a unique, secret private pair (c,F), where F is a private signature key, $F = c^e \bmod n$, and e,n is a public key of a certifying manufacturer of TMP 220.

[00035] As described in further detail below, denial of signature logic 260 provides additional functionality described below to convince or prove to a verifier platform that a private signature key held by the device was not used to generate a suspect signature during a direct proof (suspect direct signature proof), as performed by direct proof logic 250. It is appreciated that a lesser or better equipped computer than described above may be desirable for certain implementations. Therefore, the configuration of platform 200 will vary from implementation to implementation depending upon numerous factors, such

as price constraints, performance requirements, technological improvements, and/or other circumstances.

II. PLATFORM SET-UP

[00036] A "platform family" may be defined by the device manufacturer to include one or more types of platforms or devices. For instance, a platform family may be the set of all platforms (members) that have the same security relevant information. This security relevant information could contain some of the information that is included in the EK or AIK certificate in the TCPA model. It could also include the manufacturer and model number of the particular platform or device. For each platform family, a device manufacturer creates the cryptographic parameters that the manufacturer uses for that platform family. The device manufacturer creates a signature key that it uses to sign the secrets for the devices (e.g., platform 200 or TPM 220) that it manufactures as shown in FIGS. 5-6.

[00037] FIG. 5 is a flowchart illustrating a method 400 to form a platform family certificate (PFC) in accordance with one embodiment. In one embodiment, the device manufacturer utilizes a public key cryptographic function (e.g., Rivest, Shamir and Adelman (RSA) function) to create an RSA public/private key pair with public modulus n, public exponent e, and private exponent d (block 402). The public key is based on values e,n while the private key is based on d,n. This can be created using well known methods, such as those described in Applied Cryptography, by Bruce Schneier, John Wiley & Sons; ISBN: 0471117099; Second Edition (1996). In one embodiment, modulus n should be chosen large enough so that it is computationally infeasible to factor n.

[00038] The device manufacturer specifies a parameter Z, which is an integer between zero (0) and n (block 404). The device manufacturer specifies a security parameter W, which is an integer between zero (0) and n (block 406). However, picking W too small or too large may introduce a security failure. In one embodiment of the invention, W is selected to be approximately 2^{160}. Selecting W to be between 2^{80} and the square root of n is recommended. In one embodiment of the invention, the device manufacturer computes a prime number P, such that P = u*n+1 (block 408). Any value of u can be used as long as P is prime; however, to retain an acceptable level of security, the value P should be large enough so that computing a discrete logarithm "mod P" is computationally infeasible.

[00039] In one embodiment, the Direct Proof public key of the device manufacturer consists of the cryptographic parameters e,n,u,P,Z,W. These parameters will be used by a verifier to verify a direct proof signature created by a device. The device manufacturer generates a Platform Family Certificate that comprises cryptographic parameters e, n, u, P, Z, W, the security relevant information of the platform family, and the name of the device manufacturer (block 410). In one embodiment, the parameters u and P would not both be included since given n and one of these parameters, the other can be computed by $P = u*n + 1$. In one embodiment, the device manufacturer uses the same cryptographic parameters e, n, u, P, W for several different platform families, and just varies the value Z for the different platforms. In this case, the values of Z may be chosen to differ by approximately or at least 4W, although the selected difference is a design choice.

[00040] Once the Platform Family Certificate is generated, the device manufacturer provides the Platform Family Certificate to the platforms or devices it manufactures which belong to that particular platform family (block 412). The distribution of cryptographic parameters associated with the Platform Family Certificate from a prover (e.g., second platform 200 in FIG. 1) to a verifier may be accomplished in a number of ways. However, these cryptographic parameters should be distributed to the verifier in such a way that the verifier is convinced that the Platform Family Certificate was generated by the device manufacturer.

[00041] For instance, one accepted method is by distributing the parameters directly to the verifier. Another accepted method is by distributing the Platform Family Certificate signed by a certifying authority, being the device manufacturer as one example. In this latter method, the public key of the certifying authority should be distributed to the verifier, and the signed Platform Family Certificate can be given to each platform member in the platform family (prover platform). The prover platform can then provide the signed Platform Family Certificate to the verifier.

[00042] FIG. 6 is a flowchart illustrating a method 500 for the setup performed for a prover platform manufactured according to one embodiment, such as, for example, by key logic 270, as shown in FIG. 4. The TPM of the prover platform chooses a random number F such that $0 < F-Z < W$ (block 502). The TPM may blind this random number F before sending it to the certifying manufacturer for signature (block 504). This blinding operation is performed to obfuscate the exact contents of the random number F from the certifying manufacturer. In one embodiment, the TPM chooses a random value, B, where

$1 < B < n-1$ (block 506), and computes $A = B^e \bmod n$ (block 508). Then, the TPM computes $F' = F*A \bmod n$ (block 510). If the TPM does not blind F, then the TPM uses $F' = F$ and $A' = 1$ (block 512).

[00043] After performing these computations, TPM sends F' to the certifying manufacturer (block 514). The certifying manufacturer computes $c' = F'^d \bmod n$ (block 516), and provides c' to the prover (block 518). The TPM of the prover computes $c = c'*B^{-1} \bmod n$ (block 520). Notice that this implies that $c = F^d \bmod n$. The values c and F are then stored in the TPM or external storage within the prover (block 522). As described herein, F is referred to as a signature key of the TPM, whereas the secret pair c,F are referred to as cryptographic information and may also be referred to herein as a "member key". As described herein, F may be referred to as the "pseudonym exponent".

[00044] Operation of the TPM to perform a direct proof to convince a verifier that the hardware device possesses cryptographic information from a certifying manufacturer is described within co-pending U.S. Application No. 10/675,165, filed September 30, 2003. In the Direct Proof scheme, the prover's signature used in a direct proof ("direct proof signature") is validated using a public key if the platform manufacturer (issuer). Thus all members can have their signatures validated using the same public key. It can be proven that a direct proof signature created by a member does not identify which member created the direct proof signature.

[00045] To prove to a verifier that the TPM contains a unique secret pair, the TPM may obtain a value for B to use as a base according to the random base option. For example, the TPM may compute $k = B^F \bmod N$ and give B,k to the verifier in response to a signature request. Accordingly, as described herein, the value k is referred to as the "pseudonym" for the direct proof signature and B is referred to as the "base" for the direct proof signature. The TPM then constructs a direct proof signature, which is a proof that the TPM possesses F,c, such that $F = c^e \bmod n$ and $k = B^F \bmod P$, without revealing any additional information about F and c. A method for constructing a direct proof signature is given in co-pending U.S. Application No. 10/306,336, which is also owned by the assignee of the present application. TPM may use different B values each time the TPM creates a new direct proof signature so that the verifiers may not know that they received the proof from the same TPM according to the random base option.

[00046] Referring again to FIG. 4, in one embodiment, TPM 220 includes denial of signature logic 260 to handle revocation member keys. The member keys are held in hardware, but it is possible that the keys can be removed. In this case, verifiers would revoke any removed key and quit accepting direct proof signatures generated with a revoked (unknown suspect) key. As a part of the signature process, the member selects a random base B and a public key (e,n) of a certifying member to compute $k = B^F \bmod P$. The values of B and k are revealed as part of the signature. It is proven that if random bases are used, then given two different signatures, it is computationally infeasible to determine whether the two signatures were created with the same pseudonym exponent, F or different pseudonym exponents, F's.

[00047] However, if adversaries have removed the secret pseudonym exponents F's from some number of hardware devices, (say F1, F2, F3) and if a verifier has these pseudonym exponents, then the verifier can tell if a given signature was created using one of these pseudonym exponents, by checking whether $K = B^{F1} \bmod P$ or $B^{F2} \bmod P$ or $B^{F3} \bmod P$. This works for the case where the verifier has the secret F's that were removed from the hardware device. But it does not work in the case where the verifier suspects that a member key has been removed from a hardware device, but he does not have the member key, specifically the exponent F.

[00048] To give the verifier the ability to revoke a member key that he suspects is compromised, the Direct Proof methods support the named base option. In one embodiment, according to the named base option, the verifier would provide the base B, which in one embodiment, is derived from the name of the verifier. The member would use this base B in the Direct Proof signature instead of picking a random B. As long as the verifier was using the same base, the verifier could tell if two signatures sent to him used the same pseudonym exponent, F, because the two signatures would produce the same pseudonym, $B^F \bmod P$.

[00049] Thus if a verifier, using the named base option, received a direct proof signature, and suspected that the member key used to create that signature had been compromised, the verifier would be able to reject further signatures by this member key as long as he was using the same named base. However, the only way for a verifier to make effective use of the named base option is to use the same named base for a long time. This is not ideal from a privacy perspective, since it enables a verifier to link all of the transactions performed by a member with the verifier's named base.

[00050] FIG. 7 is a flowchart illustrating a method 500 performed by a verifier platform in order to verify that a cryptographic key stored within a TPM is uncompromised, in accordance with one embodiment. Representatively, at process block 510, the verifier platform determines whether it is aware of a suspect direct proof signature generated with an unknown suspect key. Suppose that the verifier platform is aware of some suspect direct proof signatures, generated with unknown suspect keys. Let B_0 be the base and K_0 be the pseudonym that was received in one of the suspect direct proof signatures. In one embodiment, the verifier platform repeats the process described below for each suspect direct proof signature.

[00051] In the embodiments described, the verifier platform does not contain a copy of the suspect key F_0 that had been used to compute $K_0 = B_0^{F0} \bmod P$. Accordingly, at process block 520, verifier platform transmits base B_0 and a pseudonym K_0 of a suspect direct proof signature, generated with the unknown, suspect key F_0. In response, verifier platform will receive one or more values from prover platform, computed using B_0 and K_0.

[00052] In one embodiment, validation of the cryptographic key stored within prover platform is formed as illustrated with reference to process blocks 540-560. The prover platform will generate a random value R. In one embodiment, the random value R is chosen in some specified interval, such as the interval between 0 and W. At process block 540, verifier platform receives the values S and T and a proof from prover platform that there exists a value R such that:

$$S = B_0^R \bmod P \text{ and } T = K_0^R \bmod P. \qquad (1)$$

[00053] In one embodiment, the received proof of the existence of the value R is in the form of a zero knowledge proof. One embodiment of such a zero knowledge proof for proving that two pairs (S,B_0) and (T, K_0) have the same discrete logarithm is given in FIG. 8. At process block 550, a verifier platform receives a proof that there exists a value F such that:

$$U = S^F \bmod P \text{ and } K = B^F \bmod P. \qquad (2)$$

[00054] Again, the proof of the existence of the value F may be performed using a zero knowledge proof. One embodiment of such a zero knowledge proof for proving that two pairs (U,S) and (K,B) have the same discrete logarithm is given in FIG. 8.

[00055] Accordingly, once verifier platform is convinced of the existence of values R and F, in one embodiment, verifier platform checks the values of U and T. If U = T

mod P, then the verifier knows that prover platform key, F was equal to the unknown, suspect key, F_0. If:

$$U \neq T \bmod P \qquad (3)$$

then the verifier knows that prover platform key, F, was not equal to the unknown, suspect key, F_0. This is easily seen since $B_0^{RF} = S^F = U \bmod P$ and $B_0^{RF0} = K_0^R = T \bmod P$. Thus $U = T \bmod P$ if and only if $F = F_0 \bmod n$.

[00056] If $U \neq T \bmod P$, prover platform key F is not equal to unknown, suspect key F_0. Accordingly, at process block 570, the verifier receives a denial that the prover signature key F was used to generate the suspect direct proof signature, referred to herein as "proving the denial of a direct proof signature". Otherwise, $U = T \bmod P$, the verifier platform receives confirmation that the prover platform was indeed using the compromised key F_0 for the direct proof signature.

[00057] In one embodiment, the prover platform denies the signature key F of the prover was used to form the suspect, direct proof signature by using a standard zero knowledge proof. As described herein, the standard zero knowledge proof for proving that two pairs have the same discrete logarithm is provided as follows. Specifically, given a set of integers k_1, h_1, k_2, h_2, and a modulus P, the zero knowledge proof will prove that there exists an e such that $k_1 = h_1^f \bmod k_2$ and $h_2^f = W^e \bmod P$ without revealing any information about f.

[00058] In one embodiment of a zero knowledge proof to show that two discrete logarithms are the same was given in co-pending U.S. Application No. 10/306,336, which is also owned by the assignee of the present application. FIG. 8 is a flow diagram 600 illustrating this zero knowledge proof. Suppose that f is in the interval between Z and Z+W. (Z could be 0, as in the case of equation 1 above.) Let $B = W * 2^{Sp + HASH\_Length}$, where Sp is a security parameter and HASH_length is the length in bits of the output of the Hash function HASH. In one embodiment Sp is chosen large enough, for example Sp = 60, so that the values of s computed below do not reveal useful information about f.

[00059] At process block 610, TPM randomly selects value t in the interval [0, B]. TPM may then compute $j_1 = h_1^t \bmod P$ and $j_2 = h_2^t \bmod P$ at process block 620. TPM may then compute $r = HASH(h_1, k_1, h_2, k_2, j_1, j_2)$ at process block 630. At process block 640, TPM may compute $s = Z+t-f*r$. Finally, at process block 650, TPM may send s, h_1, k_1, h_2, k_2, j_1, j_2 to the verifier. According to one embodiment, the verifier may then verify the proof.

[00060] FIG. 9 is a flow diagram 700 conceptually illustrating the verification of a proof that two discrete logarithms are the same, according to one embodiment. At process block 710, the challenger may compute r = HASH(h_1, k_1, h_2, k_2, j_1, j_2). The challenger may then check that $j_1 * h_1^z = k_1^r * h_1^s$ mod P and $j_2 * h_2^z = k_2^r * h_2^s$ mod P at process block 720. If the checks of process block 720 pass, the challenger may accept the proof at process block 730.

[00061] FIG. 10 is a flowchart illustrating a method 600 performed by a prover platform in response to receipt of a key validation request. As described herein, a verifier platform, once convinced of the existence of a cryptographic key stored within hardware device may verify that the stored cryptographic key is uncompromised. In accordance with one embodiment, such functionality is provided by key validation logic 260 of authentication logic 240 of TPM 220, as illustrated with references to FIGS. 2 and 3. Representatively, at process block 810, prover platform determines whether a denial of signature request is received. Once received, the functionality of process blocks 620-670 is performed.

[00062] At process block 820, verifier platform receives base B_0 and a pseudonym K_0 of a suspect signature received in a proof (suspect direct proof signature) for unknown, suspect key F_0. At process block 830, prover platform transmits computed values $S = B_O^R$ mod P, $T = K_O^R$ mod P, $U = B_O^{RF}$ mod P and $K = B^F$ mod P to the verifier. At process block 840, prover transmits a proof to verifier platform that there exists a value R such that $S = B_O^R$ mod P and $T = K_O^R$ mod P. At process block 850, prover platform transmits a direct proof to verifier platform to convince verifier platform that there exists F such that $U = S^F$ mod P and $K = B^F$ mod P.

[00063] As indicated above, in one embodiment, the proofs are performed according to the zero knowledge proof as described in FIG. 8. As also indicated above, assuming that Equation (3) evaluates to true, at process block 860, prover key F is not equal to unknown, suspect key F_0 and process block 870 is performed. At process block 870, the prover will deny that the suspect direct proof signature was generated with a signature key F of the prover platform. Otherwise, if Equation (3) evaluates to false, prover key F is equal to unknown, suspect key F_0. As a result, the prover platform would fail to prove denial of the suspect direct proof signature. Accordingly, verifier platform would fail to authenticate prover platform, since prover platform is using a compromised key.

[00064] Accordingly, one embodiment provides enhanced security capabilities to the named based option described above. However, in one embodiment, a verifier platform is prohibited from submitting to prover platforms all signatures previously received. Namely, by submitting all previously received signatures to a prover platform, a prover platform that had previously submitted a signature would be required to identify the respective signature. As a result, the verifier platform would be able to link all previous signatures from the prover platform together. In one embodiment, several methods are provided to prevent abuse of the revocation capability described by one or more embodiments herein.

[00065] In one embodiment, a prover platform is provided with a built-in capability to limit the number of signatures that the verifier can present for denial. This is a reasonable method since a very small percentage of devices will be compromised and have their keys removed. However, if more than the limit get compromised, in one embodiment, devices may be rekeyed. A device would be rekeyed only after the device had proven that it was not a compromised device. Another method is to put into the device one or more public keys (hashes of public keys) of revocation authorities. Accordingly, a verifier platform would give a denial of signature if the request for denial was approved by one of these revocation authorities. The approval could be indicated by having the revocation authority sign the request for denial, specifically to sign the pair (B_0, K_0).

[00066] In an alternate method, when a verifier asks for a signature, he gives a revocation identifier. In one embodiment, when a member is presented with a revocation identifier, the prover platform will limit signature denial to requests, including the same revocation identifier. The revocation identifier could be indicated by the low order bits of the value of B, for instance, the low order 40 bits. The verifier would indicate these low order bits of B, and the prover would use these low order bits of B, and select the rest of the bits of B randomly. The prover would then only provide a denial for signatures in which the B_0 matched these low order bits. In this way, verifier platforms could be placed into groups where two verifiers are in the same group if they used the same revocation identifier. Within a group, a verifier could tell other verifiers to reject a member key, but they could not tell verifiers outside the group to reject the member key. In one embodiment, this method may also include a limit on the number of issued denial of signature requests.

[00067] The previous application also includes a non-interactive method for Direct Proof. In addition, there have been other methods discovered for performing Direct Proof. One of these was presented by Brickell, Boneh, Chen, and Shacham and was called set signatures. Another was presented by Brickell, Camenisch, and Chen and was called Direct Anonymous Attestation. All of these methods share the property that there is a random base option such that in the creation of the signature or the interactive proof, the member creates a pseudonym, $k = B^f$ in some finite group, such as the integers modulo Q for some integer Q. Thus, the method described in this invention for proving the denial of a signature can be applied to any of these signature or interactive methods as well.

[00068] Having disclosed exemplary embodiments and the best mode, modifications and variations may be made to the disclosed embodiments while remaining within the scope of the embodiments of the invention as defined by the following claims.

CLAIMS

What is claimed is:

1. A method comprising:

convincing a verifier that a hardware device possesses cryptographic information without disclosure of the cryptographic information or any unique device identification information of the hardware device; and

convincing the verifier that the cryptographic information is uncompromised.

2. The method of claim 1, wherein convincing the verifier that the hardware device possesses the cryptographic information comprises:

performing a direct proof by the hardware device to prove that a cryptographic key is stored within the hardware device, the direct proof comprising a plurality of exponentiations, at least one being conducted using the cryptographic key of the hardware device as an exponent without exposing the cryptographic key.

3. The method of claim 1, wherein convincing a verifier that a hardware device possesses cryptographic information comprises:

using the cryptographic information to compute a pseudonym, K; and

providing the pseudonym, K, to the verifier.

4. The method of claim 3, wherein convincing the verifier that the cryptographic information is uncompromised comprises:

receiving a denial of signature request, including a base value B_0 and a pseudonym value K_0 of a suspect signature from the verifier; and

convincing the verifier that a cryptographic key stored within the hardware device used to construct the pseudonym, K, does not match an unknown, suspect key F_0 used to form the suspect signature.

5. The method of claim 1, wherein convincing the verifier that the cryptographic information is uncompromised comprises:

selecting a random exponent value R;

transmitting one or more computed values to the verifier according to a suspect-base value B_0 and a suspect pseudonym value K_0 received from the verifier, a modulus value P of the hardware device and the random exponent value R;

performing a proof by the hardware device to deny that a cryptographic key F stored within the hardware device was used to create a suspect direct proof signature, the proof comprising a plurality of exponentiations, each being conducted using one of the cryptographic key, F, the random exponent value R and other random exponent values as an exponent without exposing the cryptographic key, F, the random exponent value R and the other random exponent values.

6. The method of claim 5, wherein performing the proof comprises:

convincing the verifier that the value R exists such that:

$$S = B_0^R \bmod P \text{ and } T = K_0^R \bmod P,$$

without revealing any useful information about R; and

convincing the verifier that a value F exists such that:

$$U = S^F \bmod P \text{ and } K = B^F \bmod P,$$

without revealing any useful information about F.

7. The method of claim 5, wherein the verifier is convinced that the cryptographic key F stored within the hardware device was not used to create the suspect direct proof signature if $U \neq T \bmod P$.

8. The method of claim 1, wherein convincing the verifier that the cryptographic information is uncompromised comprises:

receiving a denial of signature request, including a suspect base value B_0 and a suspect pseudonym value K_0 of a suspect signature from the verifier;

receiving a revocation identifier associated with the suspect signature as a suspect revocation identifier; and

performing a direct proof by the hardware device to deny that a cryptographic key F stored within the hardware device matches the unknown suspect key F_0 if the suspect revocation identifier matches a revocation identifier received with a signature request from the verifier.

9. The method of claim 1, wherein convincing the verifier that the cryptographic information is uncompromised comprises:

(a) receiving a denial of signature request from the verifier, including at least one suspect direct proof signature;

(b) determining whether the request for the denial of signature has been approved by a predetermined revocation authority according to one or more public keys of one or more revocation authorities stored within the hardware device; and

(c) performing a direct proof to deny that a cryptographic key stored within the hardware device was used in a direct proof with the verifier to form the suspect direct proof signature, if the request was signed by a predetermined revocation authority.

10. The method of claim 8, further comprising:

repeating (a) - (c) for a plurality of suspect direct proof signatures; and

if the plurality of suspect direct proof signatures exceeds a suspect direct proof signature limit value, notifying the verifier that the verifier has exceeded the suspect direct proof signature limit value.

11. A method, comprising:

verifying that a hardware device possesses cryptographic information without disclosing any unique device identification information of the hardware device; and

verifying that a cryptographic key of the hardware device was not used to generate at least one suspect signature held by a verifier, where a suspect key used to generate the suspect signature is unknown to the verifier without determining any unique device identification information of the hardware device.

12. The method of claim 11, wherein prior to verifying that the hardware device possesses cryptographic information, the method comprises:

detecting compromised content of the verifier;

determining a base B_0 and a pseudonym K_0 of a suspect direct proof signature used to receive the compromised content; and storing the B_0 and a pseudonym K_0 as a suspect direct proof signature generated with an unknown, suspect key F_0.

13. The method of claim 11, wherein verifying that the hardware device possesses cryptographic information comprises:

receiving a proof from the hardware device to verify that a cryptographic key is stored within the hardware device, the proof comprising a plurality of exponentiations, at least one being conducted using the cryptographic key as an exponent without exposing the cryptographic key.

14. The method of claim 11, wherein verifying the hardware device possesses cryptographic information comprises:

computing, by the hardware device, a pseudonym, K, using the cryptographic key; and

receiving the pseudonym, K, from the hardware device.

15. The method of claim 14, wherein verifying that the cryptographic key was not used to generate the suspect signature comprises:

providing the hardware device with a denial of signature request, including a base B_0 and a pseudonym K_0 of a suspect direct proof signature generated with an unknown, suspect key F_0, the base B_0 and pseudonym K_0 having an associated revocation identifier; and

receiving a direct proof from the hardware device to convince the verifier that a cryptographic key F of the hardware device used to construct the pseudonym, K, does not match the suspect compromised key F_0 if a revocation identifier provided to the hardware device during a digital signature request matches a revocation identifier associated with the suspect direct proof signature.

16. The method of claim 11, wherein verifying that the cryptographic key was not used to generate the suspect signature comprises:

(a) providing the hardware device with a denial of signature request including a base B_0 and a pseudonym K_0 of a suspect signature formed with an unknown suspect key F_0;

(b) verifying that a cryptographic key F of the hardware device does not match the suspect compromised key F_0 without identification of the cryptographic key F of the hardware device.

17. The method of claim 16, wherein verifying further comprises:

receiving a proof from the hardware device that a value R exists such that:

$S = B_0^R \bmod P$ and $T = K_0^R \bmod P$;

without identification of any useful information about R;

receiving a proof from the hardware device that a value F exists such that:

$U = S^F \bmod P$ and $K = B^F \bmod P$,

without identification of any useful information about F; and

identifying the cryptographic key F of the hardware device as uncompromised if $U \neq T \bmod P$.

18. The method of claim 17, further comprising:

identifying the cryptographic key F of the hardware device as compromised if $U = T \bmod P$.

19. The method of claim 16, further comprising:

repeating (a) and (b) for a predetermined number of suspect direct proof signatures; and

if the predetermined number exceeds a suspect direct proof signature limit value, rekeying hardware devices that are members of a platform family defined by a certifying manufacturer of the hardware device.

20. The method of claim 11, wherein verifying that the hardware device possesses cryptographic information comprises:

transmitting a signature request to the hardware device, including a revocation identifier of a verifier of the hardware device;

receiving a digital signature of the hardware device, including the revocation identifier; and

authenticating the digital signature of the hardware device according to a public key of a manufacturer of the hardware device.

21. An apparatus, comprising:

a flash memory to store cryptographic information form a certifying manufacturer; and

a trusted platform module to convince a verifier that a hardware device possesses cryptographic information from a certifying manufacturer without disclosure of the cryptographic information or any unique device identification information of the hardware device, and to convince the verifier that the cryptographic information is uncompromised.

22. The apparatus of claim 21, wherein the trusted platform module comprises:

authentication logic to prove that a cryptographic key is stored within the hardware device according to a direct proof comprising a plurality of exponentiations, at least one being conducted using the cryptographic key as an exponent without exposing the cryptographic key.

23. The apparatus of claim 21, wherein the trusted platform module comprises:

denial of signature logic to receive a denial of signature request, including a base value B_0 and a pseudonym value K_0 of a suspect signature from the verifier and to convince the verifier that a cryptographic key stored within the hardware device and used to construct a pseudonym, K, does not match an unknown, suspect key F_0 used to form the suspect signature.

24. The apparatus of claim 21, wherein the trusted platform module comprises:

key logic to receive a unique secret pair (c,F) from a certifying manufacturer of the apparatus where F is a signature key of the hardware device of the form c^e mod P, where the pair (e, P) is a public key of the certifying manufacturer.

25. The apparatus of claim 24, wherein the trusted platform module comprises:

a flash memory to store the unique, secret pair (c,F).

26. A system, comprising:

a verifier platform coupled to a network; and

a prover platform coupled to the network, comprising:

a bus,

a processor coupled to the bus,

a chipset coupled to the bus, including a trusted platform module, in response to a challenge received over the network, the trusted module to convince a verifier that a hardware device possesses cryptographic information without disclosure of the cryptographic information or identification device information of the hardware device and to convince the verifier that the cryptographic information is uncompromised.

27. The system of claim 26, wherein the chipset comprises a graphics controller.

28. The system of claim 26, wherein the network comprises a wide area network work.

29. The system of claim 26, wherein the trusted platform module comprises:

denial of signature logic to receive a denial of signature request, including a base value B_0 and a pseudonym value K_0 of a suspect signature from the verifier, and to convince the verifier that a cryptographic key F stored within the hardware device used to compute a pseudonym, K, does not match an unknown, suspect key F_0 used to form the suspect signature.

30. The system of claim 26, wherein the trusted platform module comprises:

key logic to receive a unique secret pair (c,F) from a certifying manufacturer of the apparatus where F is a signature key of the hardware device of the form c^e mod P, where the pair (e, P) is a public key of the certifying manufacturer; and

a flash memory to store the unique, secret pair (c,F).

31. An article of manufacture including a machine readable medium having stored thereon instructions which may be used to program a system to perform a method, comprising:

convincing a verifier that a hardware device possesses cryptographic information without disclosure of the cryptographic information or any unique device identification information of the hardware device; and

convincing the verifier that the cryptographic information is uncompromised.

32. The article of manufacture of claim 31, wherein convincing a verifier that a hardware device possesses cryptographic information comprises:

using the cryptographic information to compute a pseudonym, K; and

providing that pseudonym, K, to the verifier.

33. The article of manufacture of claim 31, wherein convincing the verifier that the cryptographic key does not match the unknown, compromised key F_0 comprises:

selecting a random exponent value R;

transmitting one or more computed values to the verifier according to the suspect-base value B_0 and the suspect pseudonym value K_0 received from the verifier, a modulus value P of the hardware device and the random exponent value R;

performing a proof by the hardware device to deny that a cryptographic key F stored within the hardware device was used to create a direct proof suspect signature, the proof comprising a plurality of exponentiations, each being conducted using one of the cryptographic key, F, the random exponent value R and other exponent values as an exponent without exposing the cryptographic key, the random exponent value R and the other exponent values.

34. The article of manufacture of claim 33, wherein performing the proof comprises:

convincing the verifier that the value R exists such that:

$$S = B_0^R \bmod P \text{ and } T = K_0^R \bmod P,$$

without revealing any useful information about R; and convincing the verifier that a value F exists such that:

$$U = S^F \bmod P \text{ and } K = B^F \bmod P,$$

without revealing any useful information about F.

35. The article of manufacture of claim 34, wherein the verifier is convinced that the cryptographic key F stored within the hardware device was not used to create the suspect direct proof signature if $U \neq T \bmod P$.

36. An article of manufacture including a machine readable medium having stored thereon instructions which may be used to program a system to perform a method, comprising:

verifying that a hardware device possesses cryptographic information without disclosing any unique device identification information of the hardware device; and

verifying that a cryptographic key of the hardware device was not used to generate at least one suspect signature held by a verifier where a suspect key used to generate the suspect signature is unknown to the verifier.

37. The article of manufacture of claim 36, wherein verifying that the hardware device possesses cryptographic information comprises:

receiving a proof from the hardware device to verify that a cryptographic key is stored within the hardware device, the proof comprising a plurality of exponentiations, at least one being conducted using the cryptographic key as an exponent without exposing the cryptographic key.

38. The article of manufacture of claim 36, wherein verifying that the cryptographic key was not used to generate the suspect signature comprises:

(a) providing the hardware device with a denial of signature request including a base B_0 and a pseudonym K_0 of a suspect direct proof signature formed with an unknown suspect key F_0;

(b) verifying that a cryptographic key F of the hardware device does not match the suspect compromised key F_0 without identification of the cryptographic key F of the hardware device.

39. The article of manufacture of claim 38, wherein verifying further comprises:

receiving a direct proof from the hardware device that a value R exists such that:

$S = B_0^R \bmod P$ and $T = K_0^R \bmod P$;

without identification of any useful information about R;

receiving a direct proof from the hardware device that a value F exists such that:

$U = S^F \bmod P$ and $K = B^F \bmod P$,

without identification of any useful information about F; and

identifying the cryptographic key of the hardware device as uncompromised if $U \neq T \bmod P$.

40. The article of manufacture of claim 39, further comprising:

identifying the cryptographic key F of the hardware device as compromised if $U = T \bmod P$.

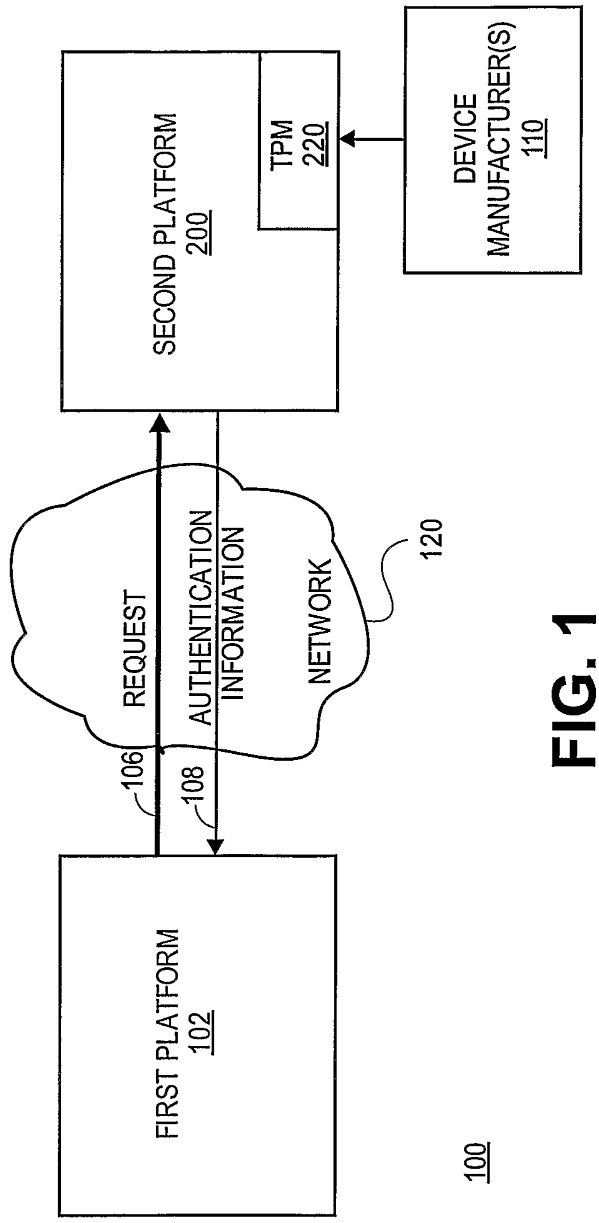
SECOND PLATFORM
200
TPM
220
DEVICE
MANUFACTURER(S)
110
REQUEST
AUTHENTICATION
INFORMATION
NETWORK
120
106
108
FIRST PLATFORM
102
100

FIG. 1

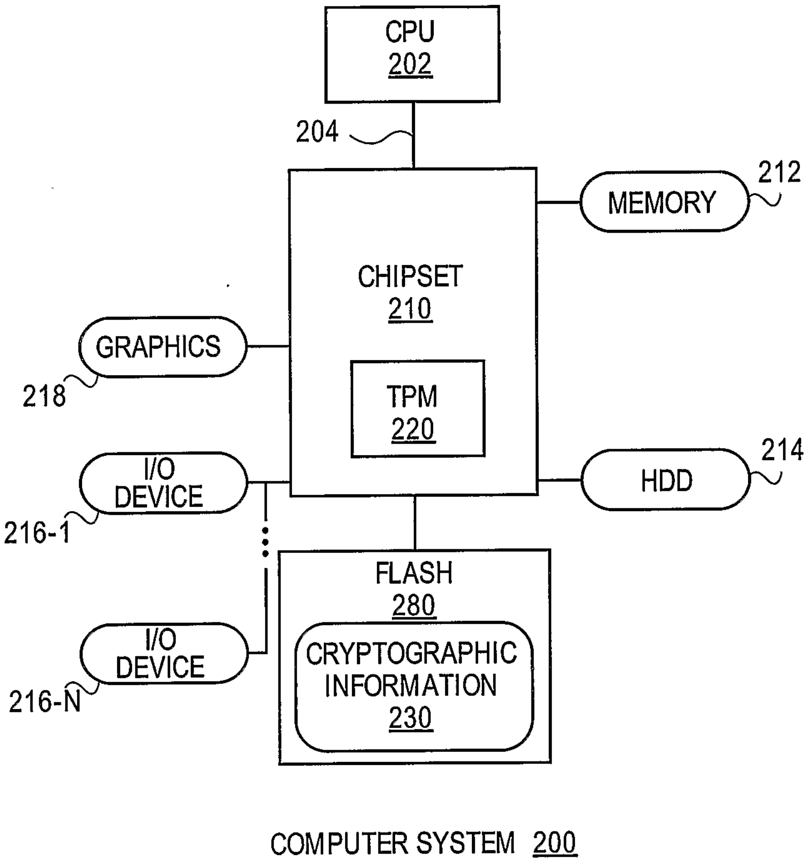
CPU
202
204
CHIPSET
210
MEMORY
212
GRAPHICS
218
TPM
220
I/O
DEVICE
216-1
HDD
214
FLASH
280
CRYPTOGRAPHIC
INFORMATION
230
I/O
DEVICE
216-N
COMPUTER SYSTEM 200

FIG. 2

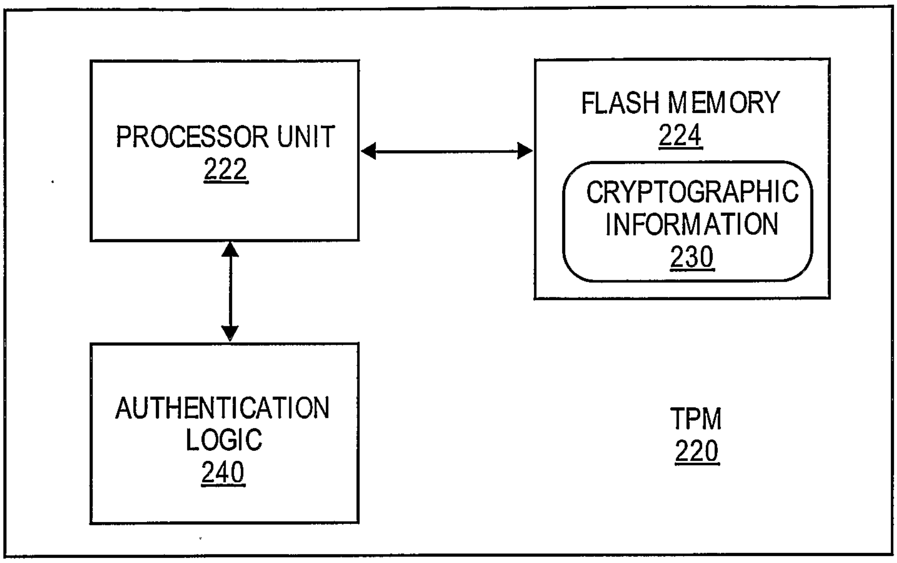
PROCESSOR UNIT
222
FLASH MEMORY
224
CRYPTOGRAPHIC
INFORMATION
230
AUTHENTICATION
LOGIC
240
TPM
220

FIG. 3

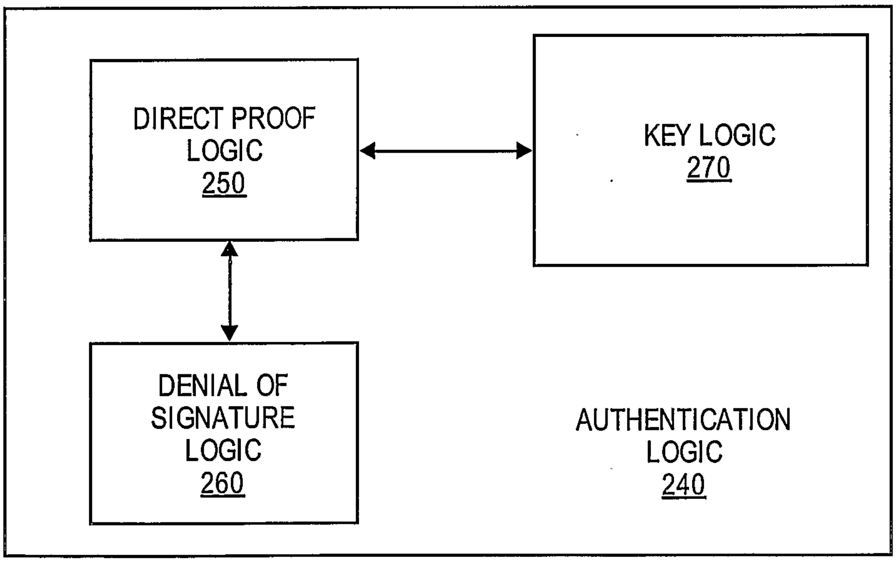
DIRECT PROOF
LOGIC
250
KEY LOGIC
270
DENIAL OF
SIGNATURE
LOGIC
260
AUTHENTICATION
LOGIC
240

FIG. 4

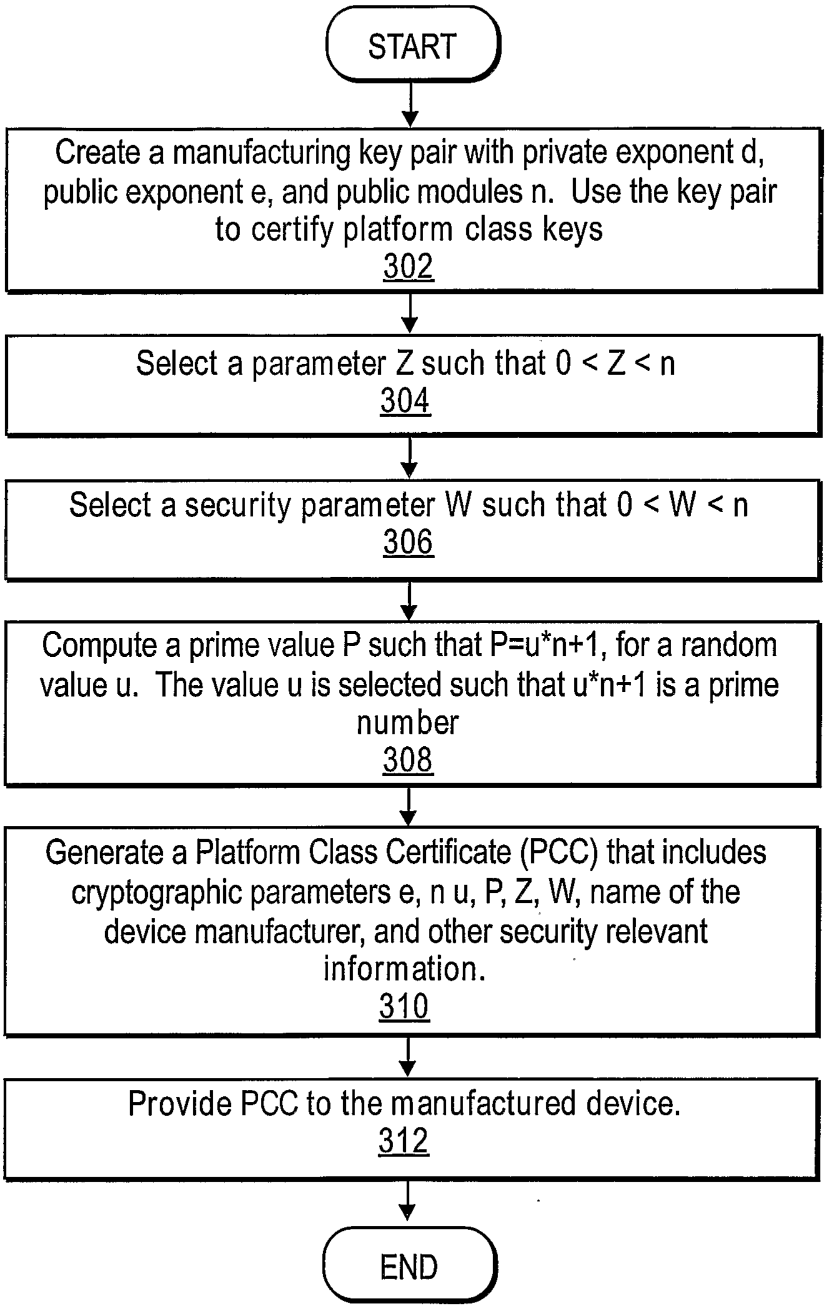
START
Create a manufacturing key pair with private exponent d, public exponent e, and public modules n. Use the key pair to certify platform class keys
302
Select a parameter Z such that 0 < Z < n
304
Select a security parameter W such that 0 < W < n
306
Compute a prime value P such that P=u*n+1, for a random value u. The value u is selected such that u*n+1 is a prime number
308
Generate a Platform Class Certificate (PCC) that includes cryptographic parameters e, n u, P, Z, W, name of the device manufacturer, and other security relevant information.
310
Provide PCC to the manufactured device.
312
END

FIG. 5

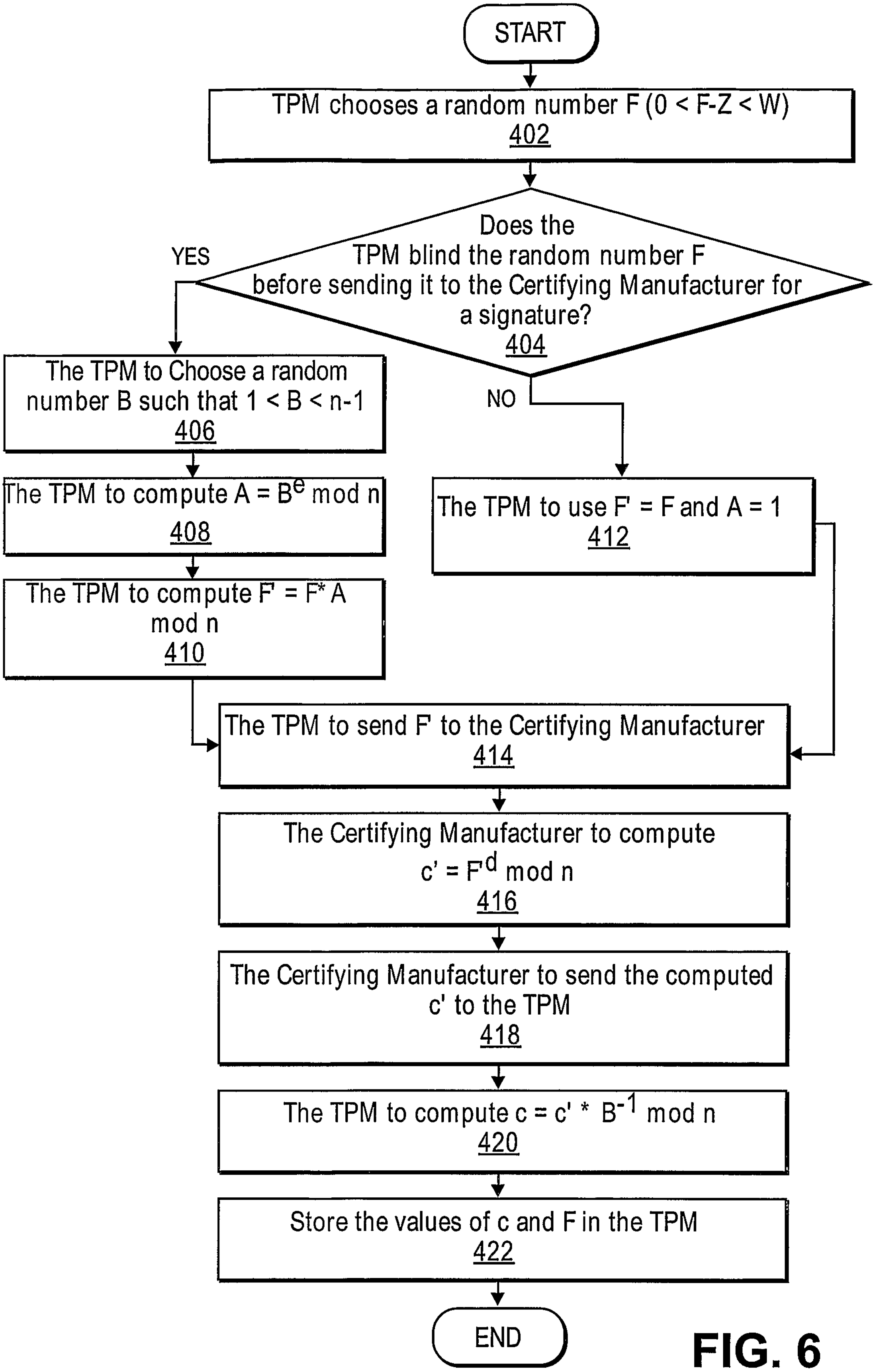
START
TPM chooses a random number F (0 < F-Z < W)
402
Does the TPM blind the random number F before sending it to the Certifying Manufacturer for a signature?
404
YES
NO
The TPM to Choose a random number B such that 1 < B < n-1
406
The TPM to compute A = B^e mod n
408
The TPM to compute F' = F* A mod n
410
The TPM to use F' = F and A = 1
412
The TPM to send F' to the Certifying Manufacturer
414
The Certifying Manufacturer to compute c' = F'^d mod n
416
The Certifying Manufacturer to send the computed c' to the TPM
418
The TPM to compute c = c' * B^-1 mod n
420
Store the values of c and F in the TPM
422
END

FIG. 6

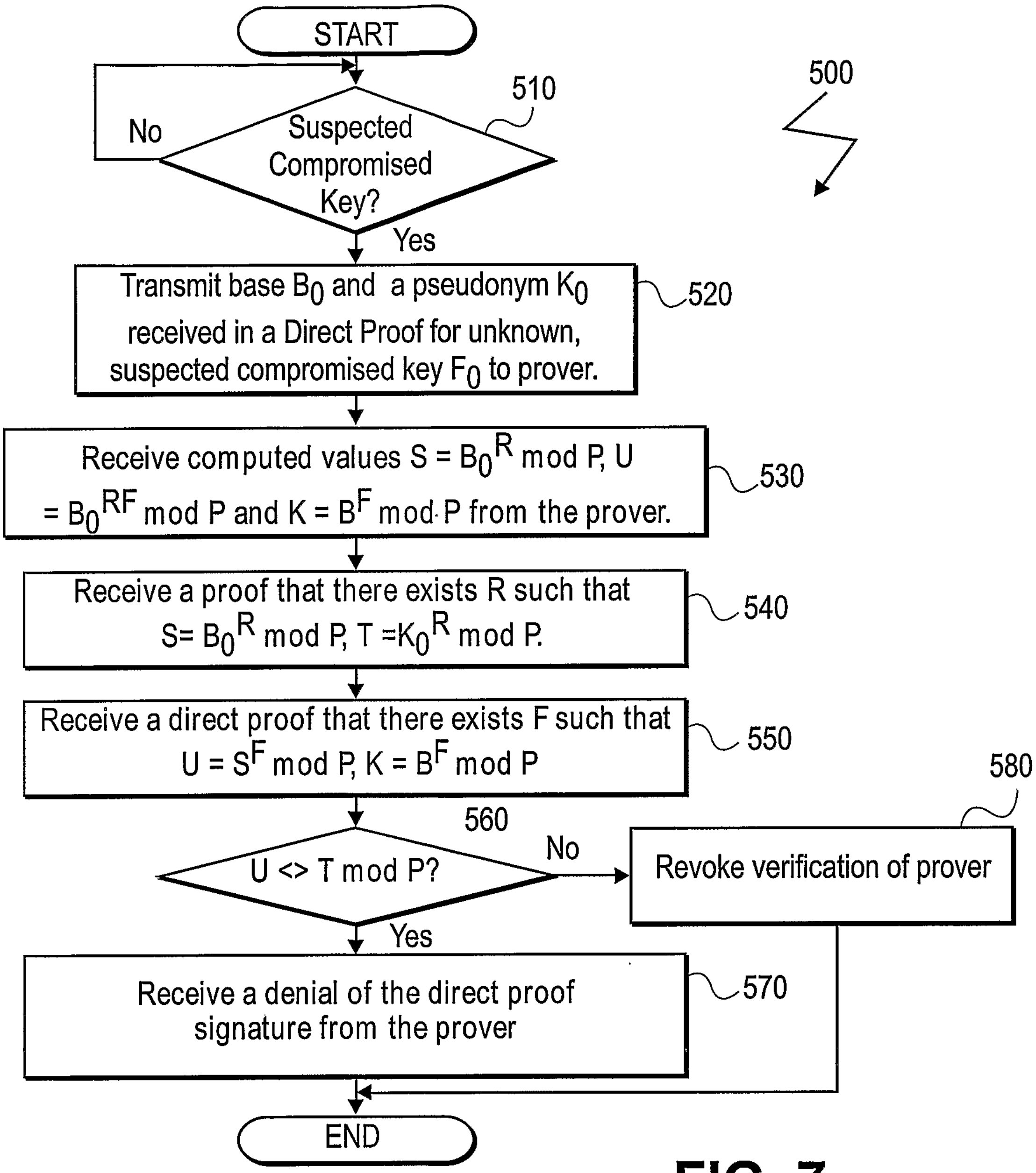
START
510
Suspected Compromised Key?
No
Yes
500
Transmit base B_0 and a pseudonym K_0 received in a Direct Proof for unknown, suspected compromised key F_0 to prover.
520
Receive computed values $S = B_0^R$ mod P, U = B_0^{RF} mod P and K = B^F mod P from the prover.
530
Receive a proof that there exists R such that S= B_0^R mod P, T =K_0^R mod P.
540
Receive a direct proof that there exists F such that U = S^F mod P, K = B^F mod P
550
560
U <> T mod P?
No
Yes
580
Revoke verification of prover
Receive a denial of the direct proof signature from the prover
570
END

FIG. 7

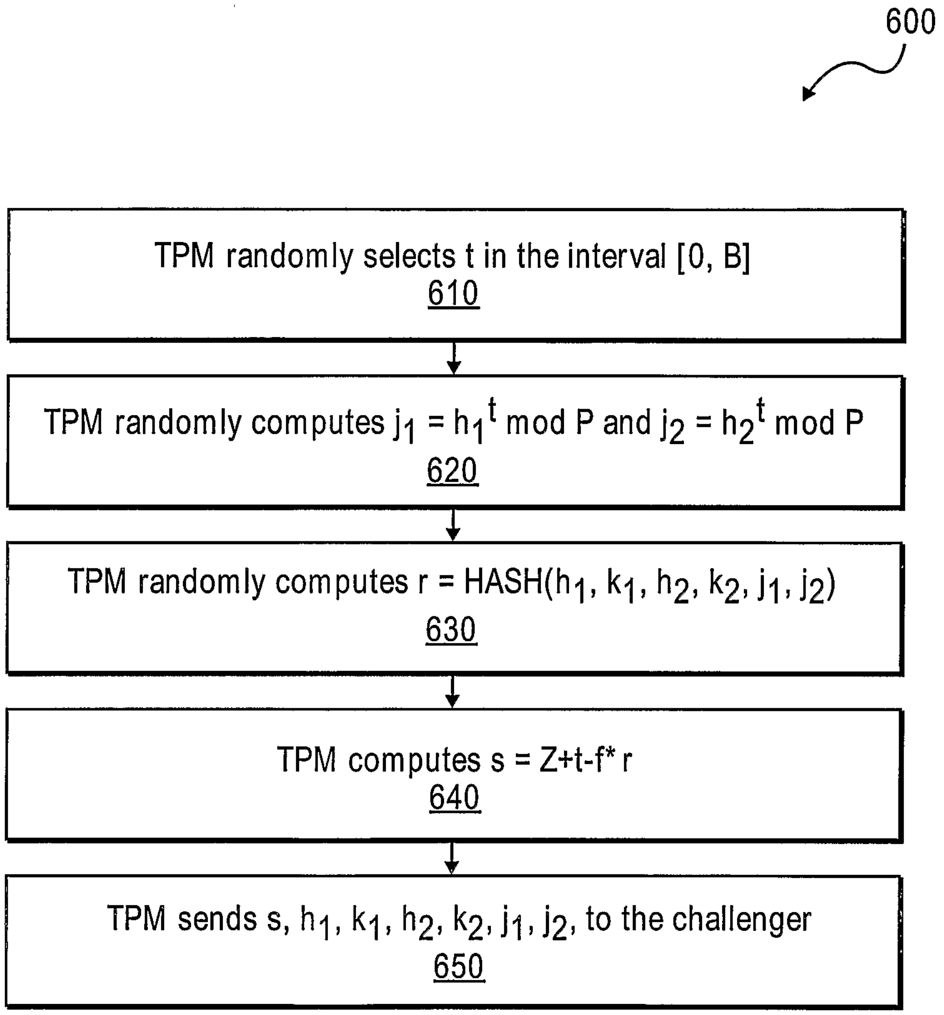
600
TPM randomly selects t in the interval [0, B]
610
TPM randomly computes $j_1 = h_1^t$ mod P and $j_2 = h_2^t$ mod P
620
TPM randomly computes r = HASH(h_1, k_1, h_2, k_2, j_1, j_2)
630
TPM computes s = Z+t-f* r
640
TPM sends s, h_1, k_1, h_2, k_2, j_1, j_2, to the challenger
650

FIG. 8

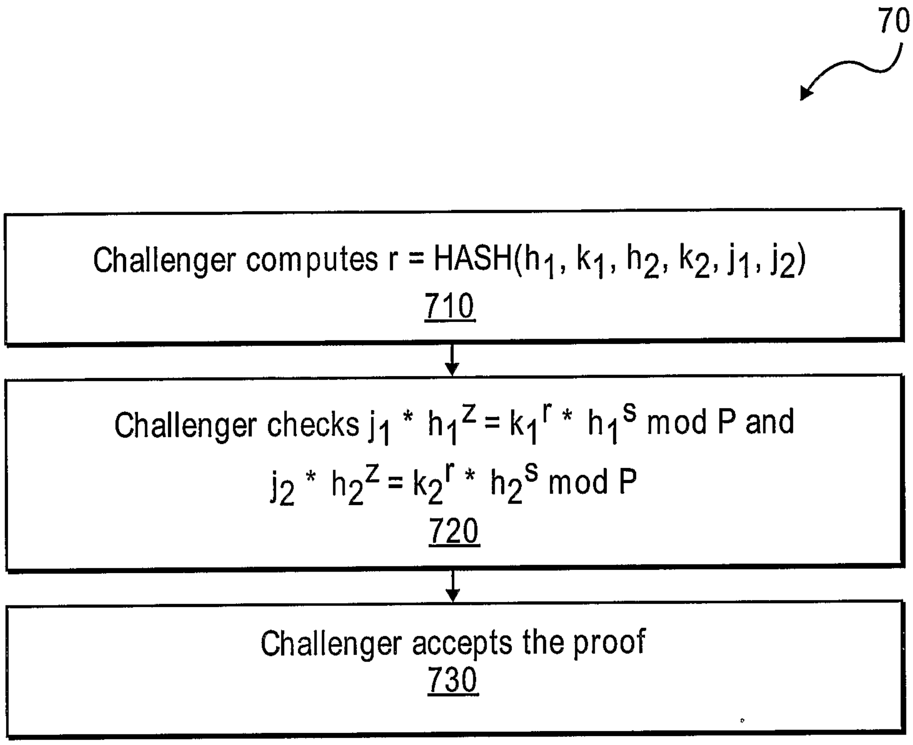
700
Challenger computes $r = \mathrm{HASH}(h_1, k_1, h_2, k_2, j_1, j_2)$
710
Challenger checks $j_1 * h_1^z = k_1^r * h_1^s \bmod P$ and
$j_2 * h_2^z = k_2^r * h_2^s \bmod P$
720
Challenger accepts the proof
730

FIG. 9

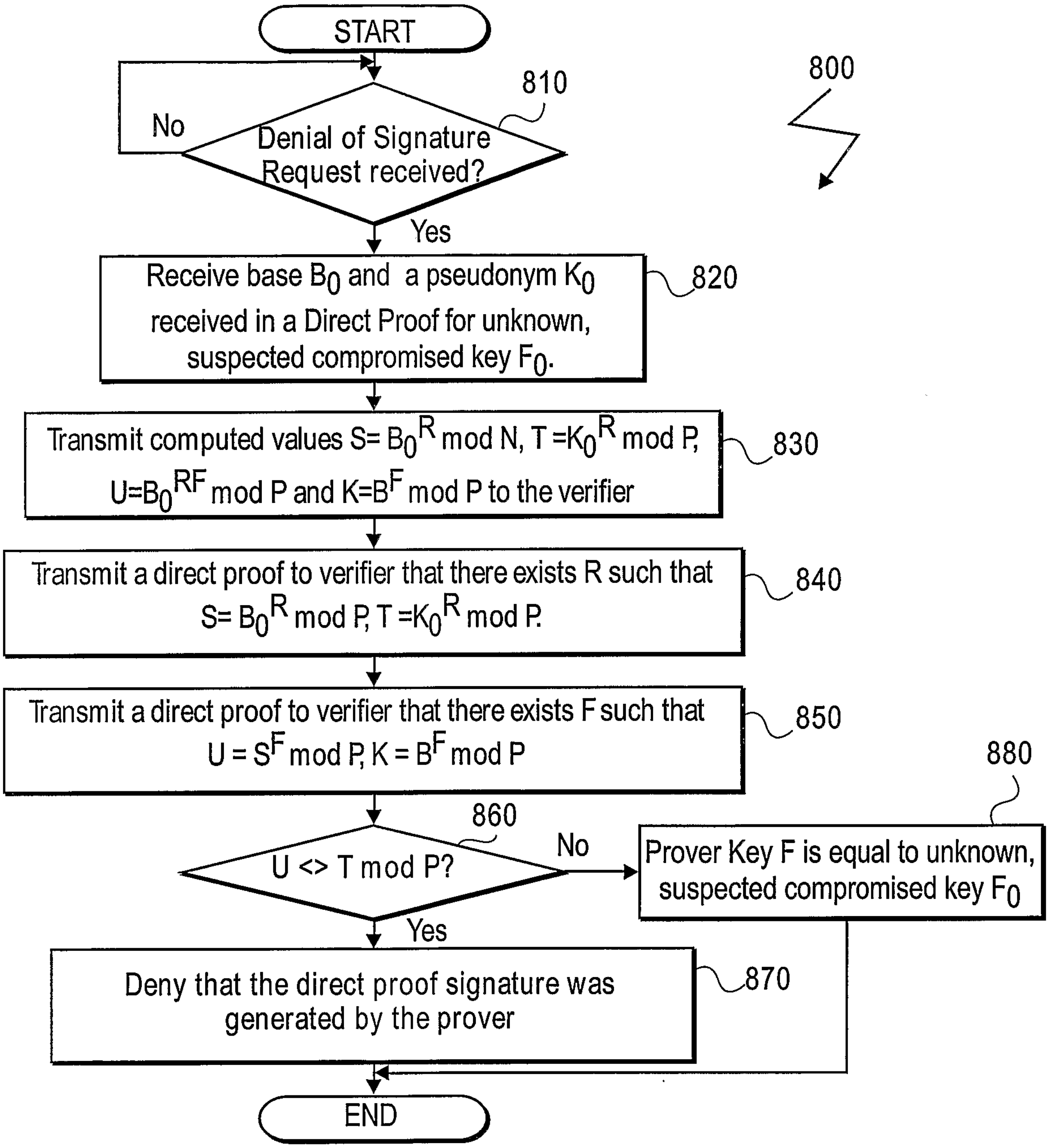
START
800
810
No
Denial of Signature Request received?
Yes
Receive base B_0 and a pseudonym K_0 received in a Direct Proof for unknown, suspected compromised key F_0.
820
Transmit computed values $S= B_0^R \bmod N$, $T = K_0^R \bmod P$, $U=B_0^{RF} \bmod P$ and $K=B^F \bmod P$ to the verifier
830
Transmit a direct proof to verifier that there exists R such that $S= B_0^R \bmod P$, $T = K_0^R \bmod P$.
840
Transmit a direct proof to verifier that there exists F such that $U = S^F \bmod P$, $K = B^F \bmod P$
850
880
860
U <> T mod P?
No
Prover Key F is equal to unknown, suspected compromised key F_0
Yes
Deny that the direct proof signature was generated by the prover
870
END

FIG. 10