PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : G06F 1/00	A1	(11) International Publication Number: WO 00/70429 (43) International Publication Date: 23 November 2000 (23.11.00)

(21) International Application Number: PCT/US00/13239

(22) International Filing Date: 15 May 2000 (15.05.00)

(30) Priority Data:
09/313,295 17 May 1999 (17.05.99) US

(71) Applicant: WAVE SYSTEMS CORP. [US/US]; Suite B200, 480 Pleasant Street, Lee, MA 01238 (US).

(72) Inventors: SPRAGUE, Steven, K.; 147 Reservoir Road, Lenox, MA 12040 (US). KAZMEIRCZAK, Gregory, J.; 36 Labaw Way, Belle Mead, NJ 08502 (US).

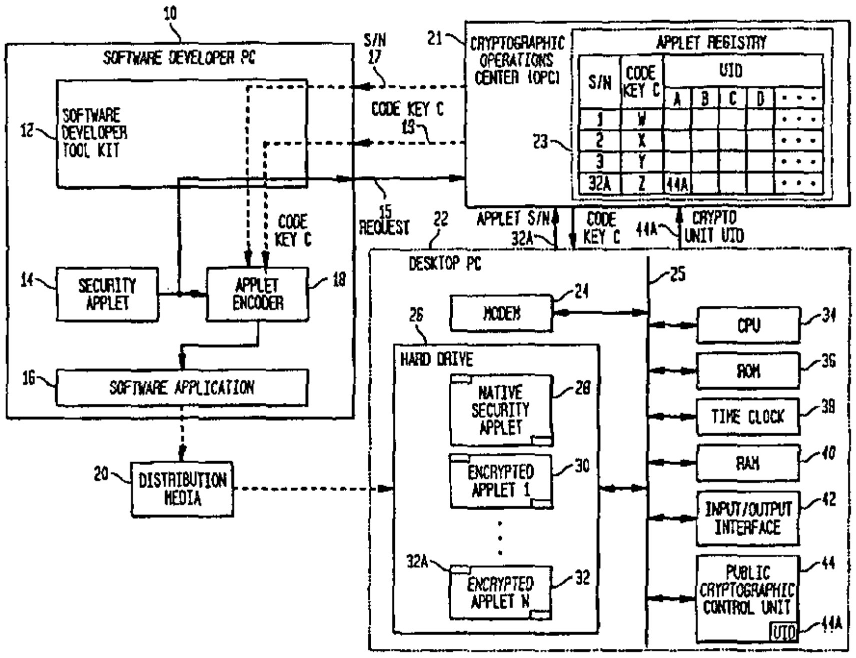
(74) Agent: JACOBSON, Allan, J.; Intellectual Property Law, 13310 Summit Square Center, Route 413 & Doublewoods Road, Langhorne, PA 19047 (US).

(81) Designated States: AU, CA, CN, JP, KR, NO, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: PUBLIC CRYPTOGRAPHIC CONTROL UNIT AND SYSTEM THEREFOR

(57) Abstract



A universally available, public cryptographic control unit (crypto unit) is used in a cryptographic system shared by multiple independent users. The crypto unit, which is installed as a peripheral device to a general-purpose computer, loads and unloads encrypted security applets into an onboard RAM memory of the crypto unit, where each security applet is run. The crypto unit and the system of which it is a part, provides a secure internal environment in which only pre-approved security applets are granted permission to load and run. The computing environment within the crypto unit is secured by a cryptographic operation center (OPC) which communicates with each crypto unit. The software developer submits a proposed security applet to the OPC prior to distributing a given security applet in order to obtain the necessary permission for the given security applet. Only if all necessary permissions are obtained from the OPC will a given security applet be allowed to load and run in the crypto unit. When a first security applet is finished running, the crypto unit unloads (swaps out) the presently loaded first security applet in encrypted form to the PC hard drive, and loads (swaps in) the next security applet. The cryptographic context of each security applet is preserved in the file stored on the PC hard drive. In such manner, a single crypto unit is shared among a plurality of independent users.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav	TM	Turkmenistan
BF	Burkina Faso	GR	Greece		Republic of Macedonia	TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	*Netherlands*	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's	NZ	New Zealand		
CM	Cameroon		Republic of Korea	PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

PUBLIC CRYPTOGRAHIC CONTROL UNIT AND SYSTEM THEREFOR

Field of the invention

The present invention relates to cryptographic systems. In particular, the present invention relates to a key management system and a shared public cryptographic control unit.

Background of the invention

Many computer applications need to perform one or more secure functions. A secure function of a computer program is a feature or operation of that computer program that is highly resistant to tampering by the user.

For example, a software program may have an expiration date after which the software program becomes inoperable. However, a typical software expiration function is not secure because it is easily defeated by resetting the local computer clock to an earlier time setting, or by modifying the software to jump over the portion of the program that checks the local computer clock.

As another example, a computer program that keeps a record of data accessed from a local encrypted database for the purpose of charging for the metered use of the local encrypted database typically has two critical registers. A first register represents the amount of past data usage, and another register represents the amount of remaining credit. However, if updating the usage and credit registers is not a secure function, the user could reduce the contents of the usage register and/or increase the contents of the credit register to defeat the system. Similarly, rented software that keeps a record of its own usage for rental charge purposes needs a secure function to prevent the user from tampering with the rental accounting registers, and other critical internal registers and functions.

As another example, a remote access database may charge authorized users for access to the database. A secure function is often needed to authenticate the identity of each user before granting access to the database. Yet another secure function is key management, i.e., the distribution of cryptographic keys to authorized users.

One class of secure function solutions is to implement secure functions in software. Implementing a secure function in software has the advantage of economy. Software implementations also have the advantage of being universal. However, implementing a software secure function in software is not as secure as implementing a secure function in hardware. On the other hand, hardware implementation of a secure function is more costly than software, and may require specialized hardware for each application. If each application requires its own specialized hardware, a hardware implementation of a secure function is not universal.

Summary of the invention

The present invention is embodied in a method and apparatus for using a cryptographic control unit as a universally available, public cryptographic control unit (crypto unit) in a system shared by multiple independent users.

The crypto unit contains a general-purpose computer processor having special purpose hardware and firmware to permit secure sharing of the crypto unit resources. In particular, the crypto unit includes a microprocessor core with a dedicated kernel of read only memory (ROM) control programming, a general purpose random access memory (RAM), a real time clock and a host input/output interface (i.e., to or from the desktop PC). In addition, the crypto unit includes a DES (Data Encryption Standard) engine, secure non-volatile storage for cryptographic keys, a signature registry RAM memory and special purpose access register.

The crypto unit is installed as a peripheral device into any general-purpose computer, such as a desktop PC. What makes the crypto unit a "public" cryptographic control unit, is that it is available to the main application program running on the PC as a secure computing resource.

In order to use the crypto unit resources, a portion of the main application program corresponding to the secure function is stored on the PC. The secure functions, which are called security applets herein, are loaded and unloaded into the onboard RAM of the crypto unit, where each security applet is run. By analogy to Java applets, which are downloaded and run inside browsers, a security applet is a portable, executable file intended to be loaded into a suitable computing entity and to perform one or more secure functions. In this sense, the crypto unit is like a special purpose coprocessor adapted for running security applications (applets).

The PC causes the security applet to be loaded into the program control memory of the crypto unit, which runs the applet and returns the result of the secure function to the PC. However, unlike a typical coprocessor, access to the crypto unit is not solely under the control of the desktop PC. That is, the desktop PC may not load and run just any security applet. The crypto unit and the system of which it is a part, provides its secure internal environment only some security applets are granted permission to load and run inside the crypto unit.

To secure the computing environment within the crypto unit, a cryptographic operations center (OPC) is provided which OPC communicates with the crypto unit. In particular, the crypto unit communicates with the OPC the first time a new security applet is encountered, and before the new security applet is allowed to run in the public crypto unit. The crypto unit also communicates with the OPC the first time a new crypto unit is installed on the desktop PC. . The crypto unit also communicates with the OPC on a regular periodic basis Furthermore, the software developer also communicates with the OPC, prior to distributing a given security applet for the purpose of obtaining the necessary permission for the given security applet load and run in the crypto unit. Only if all necessary permissions are obtained from the OPC will a given security applet be allowed to load and run in the crypto unit.

OPERATING SYSTEM

The crypto unit operating system (O/S) consists of two parts: a ROM loader control program, and a native mode security applet. The ROM loader control program is a compact dedicated kernel of control programming which is stored in ROM in the crypto unit. The native mode security applet, which may be distributed by floppy disk, CDROM or telephone modem, is a portable and writable file typically stored in the hard drive of the desktop PC.

Critical security functions are implemented in the ROM loader control program. In particular, the ROM loader control program controls the loading and unloading of security applets to and from the crypto unit and external sources, including the loading and unloading of the native mode security applet.

The native mode security applet has two main functions: to register the crypto unit at the OPC upon first use of the crypto unit, and to grant permission for the first use of each individual application security applet. As a general rule, the native mode security applet is used whenever the crypto unit communicates with the OPC.

SYSTEM OPERATION

Application developers desiring to use the public cryptographic control unit in their secure software application must first submit a proposed security applet to the OPC for consideration. The proposed security applet must meet certain standards including security standards. For example, the proposed security applet must be small enough to fit into the onboard RAM on the crypto unit. The OPC further inspects the proposed security applet for compliance with security standards.

After all security compliance tests are completed, the OPC grants or denies permission for the proposed security applet to use the crypto unit. Permission to use a proposed security applet consists of assigning a serial number and a cryptographic code key C to the approved security applet. The serial number and code key C are stored in an applet registry in the OPC. The developer uses the code key C in a process to encrypt the approved security applet, and uses the serial number to identify the encrypted security applet.

Upon start up initialization of the desktop PC, the crypto unit ROM loader control program loads the native mode security applet into onboard RAM in the crypto unit. The ROM loader control program treats the native mode security applet as though it has been previously granted permission from the OPC to load and run in the crypto unit.

The ROM loader control program facilitates the shared use of the crypto unit among multiple users. In particular, the ROM loader control program unloads (swaps out) the native mode security applet from onboard RAM in the crypto unit into the hard drive of the desktop PC to make room for loading (swaps in) a first application security applet into the onboard RAM.

The ROM loader control program then inspects the first application security applet while it is loading. After determining that the loaded first application security applet is entitled to access to the crypto unit resources, the microprocessor in the crypto unit runs the first security applet, thus turning over control of the crypto unit to the first security applet.

When the first security applet is done, the crypto unit unloads (swaps out) the presently loaded first security applet in encrypted form to the PC hard drive, and loads (swaps in) the next security applet. The cryptographic context of each security applet is preserved in the file stored on the PC hard drive. In such manner, a single crypto unit is shared among a plurality of independent users.

If an unknown application security applet is encountered (i.e., a security applet that has never been loaded into this particular crypto unit), the ROM loader control program swaps back in the native mode security applet, which establishes a secure communication session with the OPC. If the software developer who wrote the unknown application security applet has been previously granted permission by the OPC to load and run that security applet, then the crypto unit will receive from the OPC the cryptographic keys needed to decrypt and run the unknown application security applet. At the same time, the OPC records the crypto unit user identification in the applet registry, thereby associating the crypto unit with the security applet for which the OPC has granted permission to load and run. Thereafter, the crypto unit will load and unload the security applet without further communication with the OPC.

Finally, when no application security applet is running, the ROM loader control program, swaps in the native mode security applet back into the onboard RAM in the crypto unit. In such manner, each independent user uses the crypto unit for a respective separate secure application. Thus, a plurality of independent users, using a plurality of independent secure applications shares the crypto unit.

Brief description of the drawings

Figure 1 is a block diagram of a public cryptographic system in accordance with the present invention.

Figure 2 is a block diagram of a public cryptographic control unit in accordance with the present invention.

Figure 3A is a flow chart diagram illustrating a method and apparatus for generating an encrypted applet in accordance with the present invention.

Figure 3B is a flow chart diagram illustrating a method and apparatus for decrypting an encrypted applet and loading the decrypted applet into the onboard RAM of a public cryptographic control unit in accordance with the present invention.

Figure 4 is a diagram of the secure memory page format for storing an encrypted security applet in a PC hard drive memory.

Figure 5 is a flow chart diagram illustrating the process of developer registration of a security applet at the cryptographic operations center in accordance with the present invention.

Figure 6 is a flow chart diagram illustrating the process of the desktop PC initialization of a security applet at the cryptographic operations center in accordance with the present invention.

Figure 7 is a flow chart diagram illustrating the method of crypto unit initialization by the ROM loader control program portion of the O/S in accordance with the present invention.

Figure 8A is a flow chart diagram of the ROM loader control program portion of the O/S showing a method for unloading (swapping out) a security applet from the crypto unit to the PC in accordance with the present invention.

Figure 8B is a flow chart diagram of the ROM loader control program portion of the O/S showing a method for loading (swapping in) a security applet from the PC to the crypto unit in accordance with the present invention.

Figure 9A is a block diagram illustrating the method and apparatus for decrypting and loading (swap in) a cryptographic context corresponding to a security applet from the PC hard drive to the crypto unit RAM memory in accordance with the present invention.

Figure 9B is a block diagram illustrating the method and apparatus for encrypting and unloading (swap out) a cryptographic context corresponding to a security applet from the crypto unit RAM memory to the PC hard drive in accordance with the present invention.

Detailed description

SYSTEM OPERATION

A block diagram of a public cryptographic system shown in Figure 1 includes a cryptographic operations center, OPC 21, a desktop PC 22, a software developer PC 10 and a distribution media 20. The software developer uses a software developer tool kit 12 to create a security applet 14. The security applet 14 is designed to achieve a given secure function as part of a main software application 16. The software developer distributes the software application 16, included the encrypted 18 security applet 14 via some distribution media 20.

In order to encrypt the security applet 14, a software developer at PC 10 sends a request 15 over a secure communications link such as a telephone modem connection to the OPC 21. The request 15 includes the actual proposed security applet 14. In response to the request 15, the proposed security applet 14 is inspected at the OPC 21 for compliance with security standards. For example, a proposed security applet 14 should not attempt to access and output forbidden keys, tamper with the internal elapsed time counter (a secure time clock) or set permission bits (discussed below) in the crypto unit to grant itself access sensitive areas. If the proposed security applet 14 does not meet security standards for any reason, it will be denied registration.

On the other hand, if the OPC 21 approves the security applet 14 for registration, the OPC 21 will select a unique serial number (S/N) 17 and an arbitrary code key C 19 to be associated with the applet 14. The S/N 17 and code key C 19 are communicated from the OPC 21 to the software developer PC 10 over the same secure telephone modem connection as is used for the request 15. The OPC 21 retains a database of issued S/N's and corresponding issued code key C's in an applet registry 23. The proposed applet is thus officially registered and is granted permission by the OPC 21 to be used by (i.e., be run by) any public cryptographic control unit.

The software developer at PC 12, uses the received code key C in an applet encoder 18 process for encrypting the approved security applet 14. The software developer at PC 12, further uses the received S/N 17 in the applet encoder 18 process to identify the approved security applet 14. The completed security applet (encrypted using code key C 19 and identified using S/N 17) is placed in a software application 16 and distributed via some distribution media 20 such as a floppy disk, CDROM, terrestrial broadcast, satellite, cable television system or the like, to desktop PC 22.

After the software application 16 is installed at the desktop PC 22, the encrypted security applet is stored in the hard drive 26. The hard drive 26 typically holds a plurality of encrypted security applets, 28, 30, 32 which correspond to a plurality of software applications being used on the desktop PC 22. Each stored applet 32 contains an identifying S/N, such as S/N 32A. Desktop PC 22 further includes standard PC components such as a modem 24, CPU 34, ROM 36, time clock 38, RAM 40 and input/output interface 42 connected over a standard PC bus 25. In addition, the desktop PC 22 includes a crypto unit 44 having a unique unit identity (UID) 44A, which is coupled to bus 25.

In operation, the first time a software application 16 stored on the desktop PC 22 hard drive 26 requires the execution of an encrypted applet 32, the desktop 22 establishes a secure communication session with the OPC 21. The desktop PC 22 requests permission from the OPC 21 to use the encrypted applet 32. To obtain permission, the crypto unit 22 sends its UID 44A and the S/N 32A of the security applet 32 to the OPC 21.

The OPC 21 uses the previously supplied unique S/N 17 to lookup the corresponding arbitrarily supplied code key C in the applet registry 23. Also, the OPC 12 enters the transaction (use of S/N 32A by crypto unit 44) by adding to the applet registry 23. The applet registry 23 is a record of all registered security applet S/N's, the code key C that corresponds to each S/N, and all of the crypto unit UID's that have been granted permission to run each corresponding security applet. For example, the registry 23 shows that encrypted applet S/N 32A corresponding to cryptographic code key C = Z, has been registered, and that crypto unit UID = 44A has been granted permission to decrypt and execute (run) the registered applet with S/N = 32A.

PUBLIC CRYPTOGRAHIC CONTROL UNIT

A public cryptographic control unit 44 in figure 2 comprises microprocessor 206, RAM memory 222, 224 and ROM memory 208. The RAM memory is allocated to storage of a signature registry 224 and a main crypto program control 222 area. The ROM 208 contains the loader control program portion of the O/S. Also included in the crypto unit 44 is a DES engine 218, a non-volatile memory 220, an elapsed time (real time) counter 204 and an access control register 212. A host (desktop) PC interface 202 is provided for communication between the crypto unit 44 and the host PC. Communication within the crypto unit 44 is provided over a general-purpose data bus 210 carrying address and data between components within the crypto unit 44.

The DES engine 218 facilitates cryptographic operations within the protected environment of the crypto unit 44. For example, internal non-volatile memory 220 provides secure storage of cryptographic keys. The elapsed time counter 204 permits tamper proof time and date calculations within the secure environment of the crypto unit 44. Critical operations, such as reading or writing to the elapsed time counter 204, accessing or changing the contents of key storage in non-volatile memory 220, accessing or changing the contents of the signature registry 224 are restricted by hardware. In particular, access to the crypto unit is controlled by setting the individual permission bits 216 (discussed below) of the access register 212.

The access register 212 includes a special protection feature to prevent a security applet loaded in the program control RAM 222 from compromising the secure features of the crypto unit 44. In particular, the access register 212 includes a permission register, and the individual permission bits 216 of the access register 212 define which resources a given security applet will be allowed to access. For example, hardwired signals (allowance controls) 226 provide hardwired limitations as to whether a given security applet loaded in RAM 222 will be allowed to access all or part of the signature registry 224, elapsed time counter 204, and the client key and secure key storage area 220. The secret client key is unique to each crypto unit and is stored in non-volatile memory 220 at the time of manufacture as well as other cryptographic keys.

A given security applet in RAM 222 is allowed to access (read or write) critical operations only as granted permission from the access register 212. The permission register is loaded from the decrypted security applet by the loader control ROM 208 program. As a further precaution against unauthorized access, the permission register 216 may only be accessed from instruction execution in the loader control ROM 208. An address detect 214 is performed whenever the permission register is being written with a new value. In particular, only if the address detect 214 indicates that the loader control ROM is performing the permission bit loading, will the write enable signal from the address detect 214 be active. In such manner, the permission register 216 may only be loaded by the proper instruction sequence from the loader control ROM 208. Therefore, security applets running out of RAM 222 may not change the permission bits of permission register 216.

PERMISSION BITS OF THE ACCESS REGISTER

Individual permission bits 216 of the access register 212 provide control of the elapsed time counter 204. In particular, one permission bit controls whether the elapsed time counter 204 may be read, and another permission bit controls whether the elapsed time counter 204 may be written. Only the OPC, through the native mode security applet, is given permission (via the setting of a permission bit) to write a value into the elapsed time counter.

Individual permission bits 216 of the access register 212 provide control over the client key and secure key storage in non-volatile memory 220. In particular, one permission bit determines whether the applet has access to read (but not write) the client key. The client key is factory installed and may not be changed. The client key may be used in cryptographic calculations relating to a secure applet by any software developer.

Other keys stored in non-volatile memory 220 include private keys for specific software developers. That is, a given software developer may not use the shared client key. Instead, a private key may be dedicated for such given software developer. In such case, the permission bit corresponding to a dedicated private key permits security applets from the given software developer to access the dedicated private key. Security applets from other software developers will not set the permission bit for such dedicated private key, and accordingly will not have access to the dedicated private key. In addition, a separate permission bit 216 of access register 212 defines whether the loaded security applet may write a new dedicated private key over an old dedicated private key in non-volatile memory 220. In addition to dedicated private keys, non-volatile memory 220 may store digital certificates used to authenticate the public key portion of a public-private key pair.

Individual permission bits 216 of the access register 212 are used in conjunction with the signature registry portion 224 of the RAM to provide further access control as to which security applets may be loaded and unloaded into the crypto unit. In particular, setting a cancellation flag in a selected entry in the signature registry will cancel the selected security applet. The crypto chip will thereafter not load or unload a security applet designated by a cancellation flag in the signature registry. Finally, the OPC may inactivate the entire crypto unit 44 by setting an appropriate permission bit 216 that inactivates the crypto unit 44. An inactivated crypto unit 44 may not run any security applet, unless the crypto unit 44 is reactivated by the OPC.

SECURITY APPLET REGISTRATION

As indicated, application developers design security applets as part of a main application program. The security applet is written specifically to run on the crypto unit 44. The security applet must be compact enough to fit in the onboard RAM (222 in figure 2) of the crypto unit. Security applications too large to fit into the onboard RAM may be divided into two parts, i.e., into two security applets. Before a security applet can be distributed with the main application program and run on a crypto unit, the developer must register the security applet with the OPC. As indicated, the developer establishes a secure communication session with the OPC. A system suitable for secure communication with the OPC is shown in U.S. patents 5,615,264, 5,761,283 and 5,764,762.

Figure 5 shows the developer registration process at the OPC. The OPC receives a request for security applet registration at step 510. The request includes the actual proposed security applet. The OPC inspects the proposed security applet for appropriate cryptographic standards at step 512. For example, the proposed security applet may not attempt to discover the client key that is unique to each individual crypto unit, or any other secure key. There can be neither export of code nor import of additional code. Indirect program jumps are a security risk, as are indexed program loops. As a result of experience from attacks on the security of the system, numerous tests can be designed to assure that the proposed security applet is safe and properly designed. If the proposed security applet fails to pass any test, the OPC denies registration of the proposed security applet at step 512.

If all tests are passed, the OPC selects a serial number S/N and a cryptographic code key C at step 514. The OPC also enters the S/N and code key C in an applet registry (23 in figure 1) at step 514. The registration process in completed by sending the S/N and code key C for the newly registered applet to the software developer at step 516.

CRYPTOGRAPHIC CONVENTIONS USED

Figures 3A, 3B, 9A and 9B show symbols representing cryptographic operations. As used herein, the preferred process for encryption and decryption is the Data Encryption Standard (DES).

Briefly, for the electronic code book mode (ECB) of DES, an input block of 64 bits (8 bytes) is transformed into an output block of 64 bits in accordance with a 56 bit key. For decryption the reverse process is carried out, transforming 64 input bits to 64 output bits using the same 56 bit key. DES keys are typically represented in 64 bit, 8 byte quantities, with each byte having seven bits plus one parity bit, or 56 key bits plus 8 parity bits.

As used herein, performing a cryptographic operation on a variable under a secret key means to encrypt (or decrypt) that variable (usually a key) using the secret key to generate another key. Encryption may be performed under a single key, or under multiple keys, such as a triple key set. Unless otherwise indicated, encryption or decryption shall mean ECB mode of DES encryption or decryption under a triple key set. For triple key encryption, a key set of three keys (key 1, key 2, key 3) is used to encrypt a variable using DES as follows: encrypt with key 1, decrypt with key 2, and encrypt with key 3. Triple key decryption is the reverse - decrypt with key 3, encrypt with key 2, and then decrypt with key 1. CBC shall mean the cipher block chaining mode of the DES standard using an initial vector, IV. Unless otherwise stated, the IV for a CBC DES encryption or decryption shall be zero.

CRYPTO UNIT INITIALIZATION AND REGISTRATION

Figure 7 illustrates the method of crypto unit initialization and registration by the ROM loader control program. Upon powering up, the ROM loader control program (in ROM 208 of figure 2) loads an initial native mode security applet from the hard drive (26 in figure 1) into the onboard RAM (222 in figure 2) at step 710. The ROM loader control program considers the initial native mode security applet to be pre-approved and encrypted with a fixed key. The initial native mode security applet is granted access to the full resources of the crypto unit by enabling all permission bits of the access register 216. After loading, control of the crypto unit is passed to the initial native mode security applet that has just been loaded into the onboard RAM.

If this is the first time the crypto unit was used, a registration process is initiated at step 712. A secure communication session with the OPC is established at step 714, and the crypto unit enters a registration process with the OPC 716. Registration consists of entering data identifying the user (name, address, etc.) and forwarding the user data associated with the UID of the crypto unit to the OPC. During the communication session with the OPC 716 at step 714, the OPC 716 has an opportunity to download any program changes to update the initial native mode security applet. After the registration process is complete, program control is returned to the desktop PC. The crypto unit then enters a wait state until the desktop PC is ready to load the first security applet in the crypto unit to be run.

ENCRYPTION OF A REGISTERED SECURITY APPLET

Security applets are encrypted. Figure 3A is a flow chart diagram of the encryption key suite for security applet encryption. The software developer begins with the desired security applet 322. As indicated above, the security applet 322 has been previously sent to the OPC by the software developer, and an applet S/N 320 and code key C 318 have previously been received as part of the applet registration process.

The software developer selects a code key A (the programmer key) of its own choosing at step 302. Code key A is then encrypted in encryptor 304 under code key C to form encrypted code key A'. The security applet 322 is triple key CBC encrypted in encryptor 324 under code key A. A message authentication code (MAC) is calculated in encryptor 326. The MAC (also known as a manipulation detection code) is a digital signature appended to an encrypted packet that is checked by the receiver of the encrypted packet to verify that the contents of the encrypted packet have not been changed. The MAC is generated by assembling the S/N 320, code key A' and the encrypted security applet from the output of encryptor 324 into a secure packet at step 306.

The purpose of assembling a secure packet 306 is to generate a MAC 316 in encryptor 326 and append it to the secure packet to form a secure page. The developer MAC key is formed by encrypting the S/N 320 under the code key C 318 in encryptor 328. The MAC signature itself is generated by triple key CBC encrypting 326 over the secure packet 306. In particular, the last portion of the output of encryptor 326 forms the MAC signature 316, which is appended to the secure packet 306.

The computed MAC is combined with the secure packet 306 to form a secure page 308, which is outputted from the crypto unit and ultimately stored in the hard drive 26 of the host PC.

The format of the secure memory page for storing an encrypted security applet in a PC hard drive memory is shown in figure 4. The secure memory page begins with the secure packet (S/N 310 followed by the code key A' 312, followed by the encrypted security applet 314) and is terminated with the computed MAC 316.

INITIAL LOADING AND DECRYPTION OF A SECURITY APPLET

The crypto unit decrypts an initially encountered encrypted security applet as shown in the encryption key suite flow chart diagram of figure 3B. Since this is an initial loading of a security applet that has not been run before, the S/N 310 will not be found in the applet signature registry portion of RAM 224. (In the case where the S/N is found in the signature registry 224, the encrypted applet has been run before, and figure 9A will be applicable). As previously indicated, for an initially encountered security applet, the native mode security applet has sent the S/N 338 to the OPC, and received code key C 336 from the OPC.

First, the software developer code key A is recovered by decrypting code key A' 312 in decryptor 330 under code key C 336. The encrypted security applet 314 is triple key CBC decrypted in decryptor 332 under recovered code key A from the output of decryptor 330. The MAC for the secure packet (S/N 310, code key A' 312 and encrypted security applet 314) is computed in triple key CBC encryptor 340 under the developer MAC key. The developer MAC key is computed by encrypting the S/N 338 under the code key C 336 in encryptor 348, which is coupled to the key input of encryptor 340.

The computed MAC at the output of encryptor 340 is compared with the received MAC 316 in comparator 342. If the computed MAC and received MAC are equal 344, then AND gate 334 is enabled, and the decrypted security applet at the output of decryptor 332 is stored in the crypto control portion 222 of onboard RAM. However, if the computed MAC and received MAC are not equal 346, then the security applet will not be allowed to load into onboard RAM 222 and run. Instead, AND gate 334 is not enabled, and the decrypted security applet at the output of decryptor 332 is not stored in the crypto control portion 222 of onboard RAM. An error message is returned to the desktop PC.

OPC CONTROL OVER SECURITY APPLET LOADING

The present system gives the OPC control over whether a security applet can be loaded into a given crypto unit. Figure 6 illustrates the initial loading control process at the OPC. After receiving the S/N form the desktop PC at the OPC at step 610, the OPC checks whether the applet has a valid S/N at step 612. If not, the OPC returns an error message that the security applet is "INVALID". The OPC checks whether S/N, if originally valid, has since been cancelled at step 614. If so, the OPC returns an error message that the security applet has been "CANCELLED". The OPC checks whether the given crypto unit, identified by its UID, is allowed to load this particular security applet 616. If not, the OPC returns an error message that the loading of the security applet is "DISALLOWED". If the S/N is valid, not cancelled, and the crypto unit is allowed to load the security applet, code key C is looked up in the applet registry at the OPC and sent to the crypto unit at step 618.

In such manner, the OPC maintains control over initial security applet installation. For example, if a security applet has been rewritten to correct a problem, the OPC will not allow subsequent users to install the earlier version into the crypto unit. If a given crypto unit UID is known to be compromised, no further security applet loading will be allowed for that crypto unit UID.

ROM LOADER CONTROL O/S - CRYPTOGRAPHIC CONTEXT SWAPPING

The ROM loader control program (O/S) of the crypto unit supports multiple simultaneous users. To switch among users, the cryptographic context of the current security applet is unloaded from the crypto unit and stored in the hard drive of the desktop PC. Then, by retrieving a previously stored cryptographic context of a previously run security applet from the hard drive of the desktop PC, the crypto unit is restored to a previous cryptographic state corresponding to such previously run security applet. As used herein, the terms "encrypted security applet", "cryptographic context" and "encrypted security applet in (with or including) its cryptographic context" are all intended to be substantially equivalent terms.

In the present embodiment, the software developer configures the security applet to save the cryptographic parameters in the crypto program control portion 222 of onboard RAM (figure 2) before the program exits. The software developer anticipates which security parameters are needed for its security application and will be required to restore the crypto unit to its previous cryptographic state and continue the security application.

In some security applications, all of the cryptographic parameters of the crypto unit will be needed to restore the crypto unit. In other security applications, only a subset of the cryptographic parameters will be needed. In an alternate embodiment, the crypto unit automatically stores the entire cryptographic state of itself (the crypto unit) in a separate file associated with each security applet. In the latter case, the burden of switching cryptographic states (storing and restoring cryptographic contexts) is carried out automatically by operation of the crypto unit, and without intervention by the developer software.

In the present embodiment, the cryptographic context file for a given security applet includes the security applet plus the cryptographic state of the crypto unit. The format of the cryptographic context is given below:

TABLE I - CRYPTOGRAPHIC CONTEXT (29K)

Cleartext Header: Serial no. (S/N), size, revision #, time stamp
Program data - the security applet
Persistent register storage
Heap (temporary storage)
Stack
MAC/signature

Except for the cleartext header, the cryptographic context is encrypted. The cleartext header consists of the following fields:

Serial no. (S/N): The S/N is the original serial number issued to the software developer for the security applet during the registration process.

Size: Corresponds to the number of bytes in the cryptographic context to be unloaded from the crypto unit and stored in the hard drive.

Revision #: Used for tracking changes to the originally registered security applet.

Time stamp: Corresponds to the contents of the crypto unit real time clock at the time of unloading.

The encrypted portion of the cryptographic context consists of the following fields:

Program data: The security applet including any modifications made during program execution.

Heap (temporary storage): Parameters representing the cryptographic state of the crypto unit just prior to unloading.

Stack: Program stack storage such as return addresses for nested subroutines.

MAC/signature: The MAC computed over the entire cryptographic context.

TABLE II - SIGNATURE REGISTRY

The signature registry 224 portion of onboard RAM has the following format:

Serial no. (S/N)	MAC (signature)	Flags
S/N 1	MAC 1	Flag 1
S/N 2	MAC 2	Flag 2
---	---	---
S/N 31	MAC 31	Flag 3

S/N: Serial number of applet

MAC: Message authentication code for the applet cryptographic context stored in the PC hard drive.

Flags: Flags stored in the signature registry include an applet cancellation flag, which is set by the OPC to prevent any further use of the cancelled applet.

FIGURE 8A ROM LOADER CONTROL O/S - SWAP OUT

A flow chart diagram of the swap out portion of the ROM loader control program (O/S) is shown in figure 8A. The function of the swap out portion of the operating system is to unload the security applet currently running in the crypto unit, including its cryptographic context, to the hard drive of the desktop PC. For example, the security applet may have internal register storage, stack pointers and other program parameters, which are modified during execution and constitute part of its cryptographic context.

In figure 8A, when the current security applet is done at step 810, the cryptographic state of the crypto unit is saved in onboard RAM at step 812. The stored cryptographic state includes the state of DES engine (218 in figure 2) and any other variable needed to restore the crypto unit to its current condition. Then, the RAM contents are encrypted at step 814 (in accordance with the encryption key suite shown in figure 9B). The MAC for the encrypted RAM contents including the clear text header and S/N is computed at step 816, and the MAC is stored (or updated) in the RAM signature registry 224 at step 818. A secure page is assembled at step 820 and stored on the hard drive of the desktop PC at step 822.

FIGURE 8B ROM LOADER CONTROL O/S - SWAP IN

A flow chart diagram of the swap in portion of the ROM loader control program (O/S) is shown in figure 8B. The function of this portion of the operating system is to load the next security applet to the onboard RAM to run in the crypto unit, including restoring its respective previous cryptographic context, if any, from the hard drive of the desktop PC.

In figure 8B, when it is time to load a security applet into onboard RAM, the ROM loader control program first checks whether the S/N is in the signature registry portion of RAM (224 in figure 2) at step 830. The presence or absence of the S/N in the signature registry 224 determines whether or not this crypto unit has run this particular security applet before.

If the applet S/N is not in the registry, the crypto unit had not yet run this particular security applet. Then the program checks at step 834 to determine whether the signature registry is full or whether it has room for an additional entry. If the signature registry is full, an error message of "REGISTRY FULL" is returned. If the signature registry is not full, the ROM loader control program swaps in the native mode security applet at step 838, which establishes a secure communication with the OPC as described above in accordance with figure 6.

As indicated above in conjunction with figure 6, the native mode security applet sends the S/N of the proposed security applet to the OPC at step 838 and obtains a code key C at step 839. Also as indicated above, the crypto unit uses the received code key C to decrypt the proposed security applet and compute the MAC for the security applet at step 837. The decryption key suite for a security applet loaded into a given crypto unit for the first time has been described above in conjunction with figure 3B.

If the crypto unit has run this particular security applet before, then the S/N will be found in the registry at step 830. In such case, the MAC is retrieved at step 832 from the signature registry portion of RAM (224 in figure 2). The security applet is decrypted (with its cryptographic context) and the MAC is computed in step 836. The key suite for decrypting the stored applet and computing the MAC is described below in conjunction with figure 9A.

At this stage of the swap in process, there are 3 MACs associated with the security applet (with its cryptographic context) that the ROM loader control program (O/S) is attempting to load into the onboard RAM of the crypto unit. There is a first MAC retrieved from the signature registry, a second MAC received with the stored cryptographic context from the desktop PC and a third MAC computed over the incoming encrypted applet. If all 3 MACs are equal to each other at step 840, then the decrypted security applet is loaded into the crypto program control portion of RAM, and execution of the security applet is begun at step 842. Otherwise, an error message of "ACCESS DENIED" is returned to the PC from step 840.

CRYPTOGRAPHIC CONTEXT FILES

Figure 9A (swap in) and figure 9B (swap out) show the respective decryption and encryption key suites for swapping security applets (in respective cryptographic contexts) between the crypto program control portion of RAM 222 and the hard drive 26 on the desktop PC. In particular, figure 9A is a block diagram illustrating the method and apparatus for decrypting and loading (swap in) a cryptographic context corresponding to a security applet from the PC hard drive to the crypto unit RAM memory. Figure 9B is a block diagram illustrating the method and apparatus for encrypting and unloading (swap out) a cryptographic context corresponding to a security applet from the crypto unit RAM memory to the PC hard drive.

CRYPTOGRAPHIC CONTEXT SWAP OUT - FIGURE 9B

In figure 9B, the contents of the crypto program control portion of RAM 222 are to be unloaded as an encrypted file 962A in hard drive 26. The signature registry portion 224 of RAM memory is not unloaded. The various encryption keys generated are based on a first fixed string A 940, a second fixed string B 956 and a secret key, called the client key 942. The client key 942 is stored in a programmable memory (220 in figure 2). The client key memory 942 is typically non-volatile, and may be implemented by any suitable non volatile memory, such as fuseable link, EEPROM, battery backed up RAM and the like. The stored client key 942 is unique to each crypto unit and is installed at the time of manufacture.

First fixed string A 940 is encrypted under the client key 942 in encryptor 944. The output of encryptor 944 is used as the key in encryptor 946 to encrypt the S/N (cleartext) of the applet to be unloaded. The output of encryptor 946 is used as the key to encrypt the security applet in triple key CBC encryptor 948. Note that the encryption key (to encryptor 948) for the security applet swap out is not the same key as was used for initial loading of the security applet. For initial loading of the security applet, the key used was the developer code key A. In figure 9B, the key used for unloading is a function of fixed string A 940, the S/N and the client key 942. Since each client key is unique to each crypto unit, the swapped out cryptographic context stored in the hard drive 26 may not be swapped back into another crypto unit. That is, once a security packet has been swapped out of a crypto unit to the hard drive 26 using one client key, the swapped out security packet (in its cryptographic context) cannot be loaded into a different crypto unit having a different client key.

To generate a MAC for the cryptographic context (which includes the security applet), a secure packet 950 is assembled. The secure packet 950 consists of the S/N in the clear and the encrypted security applet (with its cryptographic context). The MAC is generated by triple key CBC encrypting the secure packet under a key derived from the output of encryptor 954. As can be seen from figure 9B, the MAC key output from encryptor 954 is a function of fixed string B 956, (and via encryptors 944 and 946) the S/N, the client key 942 and fixed string A 940.

In particular, the output of encryptor 946 is input as the encryption key to encryptor 954, which encrypts fixed string B to be the MAC key to triple key CBC encryptor 952. The MAC at the output of encryptor 952 is assembled along with the secure packet to form a secure page 958. The secure page 958 is stored 962A in the hard drive 26 along with other cryptographic contexts 962N as well as the cryptographic context of the swapped out native mode security applet 960.

CRYPTOGRAPHIC CONTEXT SWAP IN - FIGURE 9A

When the crypto unit switches between multiple simultaneous security applications, a previously stored cryptographic context 912, 918A - 918N in figure 9A is loaded from the hard drive 26 to the crypto program control portion 222 of on board RAM. The crypto unit uses the contents of the signature registry 224 to determine whether each of the previously stored cryptographic contexts 912, 918A to 918N will be allowed to load and run. The native mode security applet in its cryptographic context 912 is swapped in and out of the crypto unit in the same manner as the other multiple simultaneous security applets 918A - 918N run by the crypto unit.

The key suite of figure 9A (swap in) carries out the reverse cryptographic process of the key suite in figure 9B (swap out). In particular, fixed string A 910 is encrypted under the client key 914 in encryptor 916. The output of encryptor 916 is used as the key in encryptor 920 to encrypt the S/N of the security applet and cryptographic context 918A to be loaded. The output of encryptor 920 is used as the applet decryption key to decrypt the security applet cryptographic context in triple key CBC decryptor 922. The applet decryption key (to decryptor 922) for the security applet swap in is the same key as was used to encrypt the security applet during swap out.

The MAC key for the cryptographic context 918A is computed by first encrypting fixed string B 930 under the applet decryption key (output of encryptor 920) in encryptor 932. The output of encryptor 932 is then used as the key in encryptor 926 to form a computed MAC over the secure page portion of the cryptographic context 918A. To check the MAC, the stored MAC from the signature registry portion 224 of RAM is retrieved. Then, all three of the computed MAC from the output of encryptor 926, the stored MAC from the signature registry 224 and the received MAC from the cryptographic context 918A are compared in comparator 928. If all three MACs are equal at step 934, then AND gate 924 is enabled to load the received security applet into the crypto program control portion 222 of RAM. If any one of the three MACs are not equal to the others at step 936, then AND gate 924 is not enabled to load the received security applet into the crypto program control portion 222 of RAM.

SECURITY APPLET SWAPPING

Security applets may be swapped into the crypto unit by either a one pass or a two pass process. A two pass process has been described above. That is, the ROM loader control program inspects an encrypted security applet before loading it into the crypto program control portion of onboard RAM. In a two pass implementation, if all MAC signature tests are passed, the security applet is then decrypted and loaded into onboard RAM in a second pass. If loading is disallowed on the first pass, no portion of the security applet will be loaded into onboard RAM on the second pass.

In a one pass implementation, the ROM loader control program inspects an encrypted security applet while simultaneously decrypting and loading the decrypted security applet into the crypto program control portion of onboard RAM. If the MAC signature test fails (step 840 in figure 8B) control over the crypto unit is not passed to the just loaded security applet. Instead, the next security applet or the native mode security applet is loaded into the crypto program control portion of onboard RAM overwriting the previously loaded disallowed security applet. However, if the MAC signature test is passed, then the ROM loader control program passes control over the crypto unit to the just loaded security applet.

A two pass embodiment is generally more secure, because no portion of the new security applet is loaded into crypto program control portion of RAM 222 before all MAC signature tests are preformed. A one pass embodiment generally results in faster security applet swapping because the new security applet begins execution in the onboard RAM without waiting for a second pass.

CRYPTO UNIT SUPERVISION BY THE OPC

The crypto unit 44 in figure 1 is periodically supervised by the OPC 21. That is, at least once per month, or at any other selected time interval, the crypto unit 44 initiates a communication session with the OPC 21. Communication may be via modem 24 to a dial up connection or via the TCP/IP protocol over an Internet connection. In either case, the state of the crypto unit 44 is reported to the OPC 21. The purpose of the periodic communication is to synchronize the contents of the crypto unit 44 with what is expected at the OPC 21.

For example, during periodic communication with the OPC 21, the elapsed time counter 204 (figure 2) is checked against its expected value and synchronized if necessary. Any wide discrepancy of elapsed time may be an indication of tampering, and may result in inactivation of the crypto unit by the OPC. The OPC can set one or more of the permission bits 216 in access register 212 to inactivate a crypto unit. Once inactivated, an inactive crypto unit may not load or run any security applet.

Also, during periodic communication with the OPC 21, the signature registry (224 in figure 2) is checked to review which security applets have been loaded and run in that crypto unit. If a security applet has since been cancelled (i.e., system wide permission to run that security applet has been withdrawn), the cancellation flag corresponding to that security applet will be set in the signature registry. Thereafter, in conjunction with the allowance controls (226 in figure 2), the crypto unit 44 will not swap in (load) the cancelled security applet.

What is claimed is:

1. In a cryptographic key distribution system, including a user computer having a cryptographic control unit, a software developer computer and a cryptographic operations center, a method comprising:

generating a first security applet at said software developer computer;

transmitting said first security applet from said software developer computer to said cryptographic operations center;

receiving a first cryptographic key from said cryptographic operations center at said software developer computer;

receiving a first serial number from said cryptographic operations center at said software developer computer;

using said first cryptographic key in a process to encrypt said first security applet to form a first encrypted security applet;

appending said first serial number to said first encrypted security applet to form a first secure packet; and

distributing said first secure packet to said user computer.

2. A method in accordance with claim 1, wherein said cryptographic control unit includes a program control memory, said method further comprising:

transmitting said first serial number from said cryptographic control unit to said cryptographic operations center;

receiving said first cryptographic key from said cryptographic operations center at said cryptographic control unit;

using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet; and

loading said first security applet in said program control memory.

3. A method in accordance with claim 1, wherein said step of using said first cryptographic key in a process to encrypt said first security applet to form a first encrypted security applet further comprises:

encrypting said first security applet at said software developer computer under a programmer key to form a first programmer encrypted security applet;

encrypting said programmer key at said software developer computer under said first cryptographic key to form a first encrypted programmer key;

appending said first encrypted programmer key and said first programmer encrypted security applet to form said first encrypted security applet.

4. A method in accordance with claim 2, wherein said software developer computer further includes a programmer key encrypted under said first cryptographic key to form a first encrypted programmer key in said first encrypted security applet, and said step of using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet at said user computer comprises:

receiving said first secure packet including said first programmer encrypted security applet at said user computer;

decrypting said first encrypted programmer key at said user computer under said first cryptographic key to form a recovered programmer key; and

decrypting said first programmer encrypted security applet under said recovered programmer key to recover said first security applet.

5. A method in accordance with claim 1, further comprising:

storing a security applet registry table recording the correspondence between a plurality of security applets, cryptographic keys and serial numbers, said security applet registry table having an entry indicating that said first serial number and said first cryptographic key correspond to said first security applet.

6. A method in accordance with claim 2, wherein said user computer includes a first user identification number, said method further comprising:

transmitting said first user identification number from said user computer to said cryptographic operations center;

storing a security applet registry table recording the correspondence between a plurality of security applets, cryptographic keys, serial numbers and user identification numbers, said security applet registry table having an entry indicating that said first serial number, said first cryptographic key and said first user identification number correspond to said first security applet.

7. A method in accordance with claim 2, wherein said user computer further includes a user computer hard drive memory, and said system further includes a second software developer computer, a second security applet having a respective second serial number and second cryptographic key corresponding thereto, said method further comprising:

encrypting the contents of said user control memory in a process using a first user computer key to form a first encrypted security context;

storing said first encrypted security context on said user computer hard drive memory; and

loading said second security applet in said program control memory.

8. A method in accordance with claim 7, further comprising:

encrypting the contents of said user control memory in a process using a second user computer key to form a second encrypted security context;

storing said second encrypted security context on said user computer hard drive memory;

decrypting said first encrypted security context in a process using said first user computer key to recover said first security context; and

loading said first security context in said program control memory.

9. In a cryptographic key distribution system, including a user computer having a cryptographic control unit, a software developer computer and a cryptographic operations center, an apparatus comprising:

means for generating a first security applet at said software developer computer;

means for transmitting said first security applet from said software developer computer to said cryptographic operations center;

means for receiving a first cryptographic key from said cryptographic operations center at said software developer computer;

means for receiving a first serial number from said cryptographic operations center at said software developer computer;

means for using said first cryptographic key in a process to encrypt said first security applet to form a first encrypted security applet;

means for appending said first serial number to said first encrypted security applet to form a first secure packet; and

means for distributing said first secure packet to said user computer.

10. An apparatus in accordance with claim 9, wherein said cryptographic control unit includes a program control memory, said apparatus further comprising:

means for transmitting said first serial number from said cryptographic control unit to said cryptographic operations center;

means for receiving said first cryptographic key from said cryptographic operations center at said cryptographic control unit;

means for using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet; and

means for loading said first security applet in said program control memory.

11. An apparatus in accordance with claim 9, wherein said means for using said first cryptographic key in a process to encrypt said first security applet to form a first encrypted security applet further comprises:

means for encrypting said first security applet at said software developer computer under a programmer key to form a first programmer encrypted security applet;

means for encrypting said programmer key at said software developer computer under said first cryptographic key to form a first encrypted programmer key;

means for appending said first encrypted programmer key and said first programmer encrypted security applet to form said first encrypted security applet.

12. An apparatus in accordance with claim 10, wherein said software developer computer further includes a programmer key encrypted under said first cryptographic key to form a first encrypted programmer key in said first encrypted security applet, and said means for using said first

cryptographic key in a process to decrypt said first security applet from said first programmer encrypted security applet at said user computer comprises:

means for receiving said first secure packet including said first programmer encrypted security applet at said user computer;

means for decrypting said first encrypted programmer key at said user computer under said first cryptographic key to form a recovered programmer key; and

means for decrypting said first programmer encrypted security applet under said recovered programmer key to recover said first security applet.

13. An apparatus in accordance with claim 9, further comprising:

means for storing a security applet registry table recording the correspondence between a plurality of security applets, cryptographic keys and serial numbers, said security applet registry table having an entry indicating that said first serial number and said first cryptographic key correspond to said first security applet.

14. An apparatus in accordance with claim 10, wherein said user computer includes a first user identification number, said apparatus further comprising:

means for transmitting said first user identification number from said user computer to said cryptographic operations center;

means for storing a security applet registry table recording the correspondence between a plurality of security applets, cryptographic keys, serial numbers and user identification numbers, said security applet registry table having an entry indicating that said first serial number, said first cryptographic key and said first user identification number correspond to said first security applet.

15. An apparatus in accordance with claim 10, wherein said user computer further includes a user computer hard drive memory, and said system further includes a second software developer computer, a second security applet having a respective second serial number and second cryptographic key corresponding thereto, said apparatus further comprising:

means for encrypting the contents of said user control memory in a process using a first user computer key to form a first encrypted security context;

means for storing said first encrypted security context on said user computer hard drive memory; and

means for loading said second security applet in said program control memory.

16. An apparatus in accordance with claim 15, further comprising:

means for encrypting the contents of said user control memory in a process using a second user computer key to form a second encrypted security context;

means for storing said second encrypted security context on said user computer hard drive memory;

means for decrypting said first encrypted security context in a process using said first user computer key to recover said first security context; and

means for loading said first security context in said program control memory.

17. In a cryptographic key distribution system, including a user computer having a cryptographic control unit, a software developer computer and a cryptographic operations center, said software developer computer generating a first security applet and encrypting said first security applet in process using a first cryptographic key to form a first encrypted security applet and distributing said first encrypted security applet to said user computer, a cryptographic key distribution method at said cryptographic operations center comprising:

receiving said first security applet from said software developer computer at said cryptographic operations center;

transmitting a first serial number from said cryptographic operations center to said software developer computer;

transmitting a first cryptographic key from said cryptographic operations center at said software developer computer;

receiving said first serial number from said cryptographic control unit at said cryptographic operations center;

transmitting said first cryptographic key from said cryptographic operations center to said cryptographic control unit.

18. A method in accordance with claim 17, wherein said cryptographic control unit includes a program control memory, said method further comprising:

receiving said first encrypted security applet at said cryptographic control unit including said first serial number;

transmitting said first serial number from said cryptographic control unit to said cryptographic operations center;

receiving said first cryptographic key from said cryptographic operations center at said cryptographic control unit;

using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet; and

loading said first security applet in said program control memory.

19. In a cryptographic key distribution system, having a software developer computer and a cryptographic operations center, said software developer computer generating a first security applet identified by a first serial number, and encrypting said first security applet in process using a first cryptographic key to form a first encrypted security applet, said system further including a user computer having a cryptographic control unit with a program control memory, a method comprising:

receiving said first security applet including said first serial number at said cryptographic control unit;

transmitting said first serial number to said cryptographic operations center;

receiving said first cryptographic key from said cryptographic operations center at said cryptographic control unit;

using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet; and

loading said first security applet in said program control memory.

20. In a cryptographic key distribution system, including a user computer having a cryptographic control unit, a software developer computer and a cryptographic operations center, said software developer computer generating a first security applet and encrypting said first security applet in process using a first cryptographic key to form a first encrypted security applet and distributing said first encrypted security applet to said user computer, a cryptographic key distribution apparatus at said cryptographic operations center comprising:

means for receiving said first security applet from said software developer computer at said cryptographic operations center;

means for transmitting a first serial number from said cryptographic operations center to said software developer computer;

means for transmitting a first cryptographic key from said cryptographic operations center at said software developer computer;

means for receiving said first serial number from said cryptographic control unit at said cryptographic operations center;

means for transmitting said first cryptographic key from said cryptographic operations center to said cryptographic control unit.

21. An apparatus in accordance with claim 20, wherein said cryptographic control unit includes a program control memory, said apparatus further comprising:

means for receiving said first encrypted security applet at said cryptographic control unit including said first serial number;

means for transmitting said first serial number from said cryptographic control unit to said cryptographic operations center;

means for receiving said first cryptographic key from said cryptographic operations center at said cryptographic control unit;

means for using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet; and

means for loading said first security applet in said program control memory.

22. In a cryptographic key distribution system, having a software developer computer and a cryptographic operations center, said software developer computer generating a first security applet identified by a first serial number, and encrypting said first security applet in process using a first cryptographic key to form a first encrypted security applet, said system further including a user computer having a cryptographic control unit with a program control memory, an apparatus comprising:

means for receiving said first security applet including said first serial number at said cryptographic control unit;

means for transmitting said first serial number to said cryptographic operations center;

means for receiving said first cryptographic key from said cryptographic operations center at said cryptographic control unit;

means for using said first cryptographic key in a process to decrypt said first security applet from said first encrypted security applet; and

loading said first security applet in said program control memory.

FIG. 1

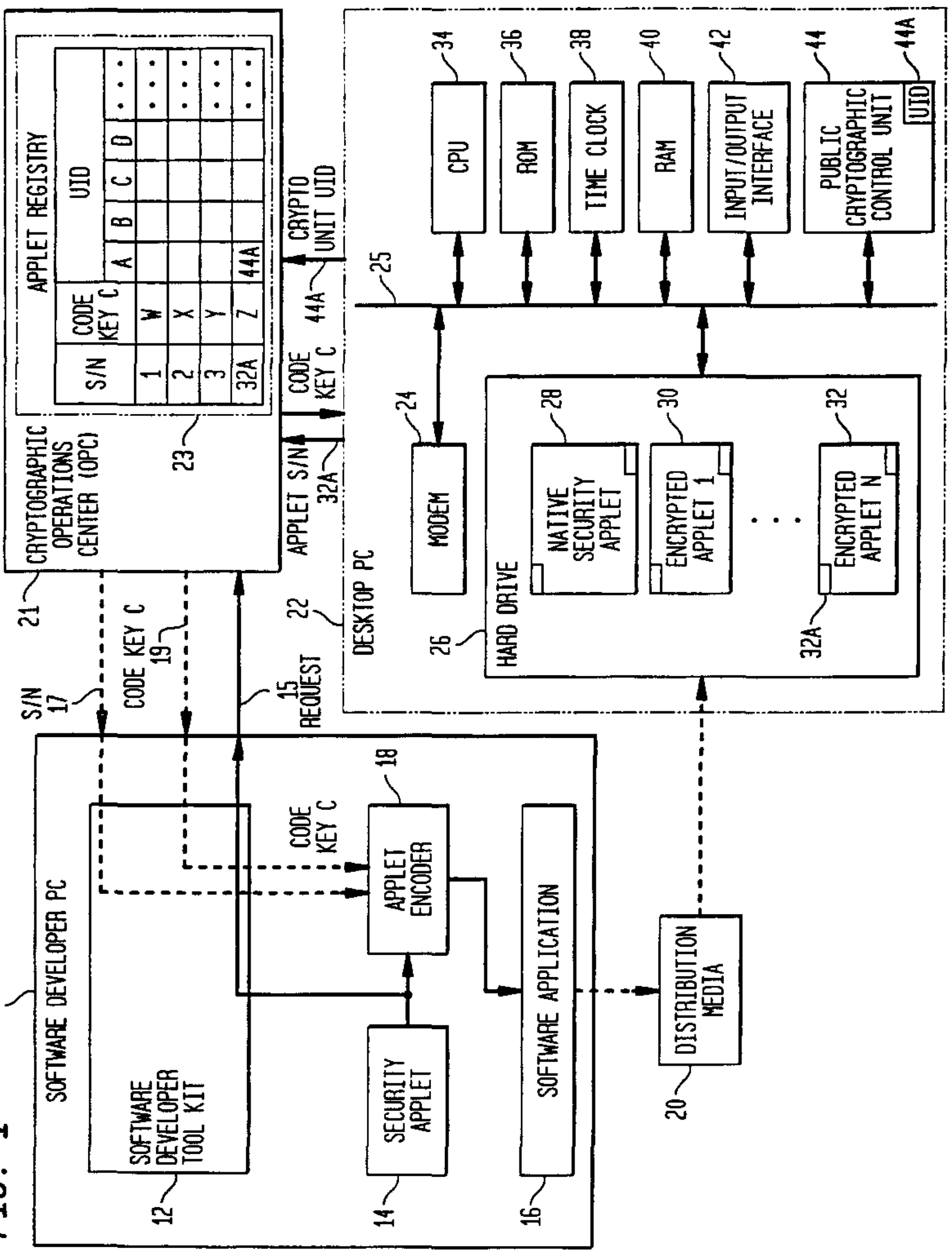

SOFTWARE DEVELOPER PC
10
SOFTWARE DEVELOPER TOOL KIT
12
SECURITY APPLET
14
SOFTWARE APPLICATION
16
APPLET ENCODER
18
CODE KEY C
DISTRIBUTION MEDIA
20
S/N 17
CODE KEY C 19
15 REQUEST
CRYPTOGRAPHIC OPERATIONS CENTER (OPC)
21
23
APPLET REGISTRY
S/N
CODE KEY C
UID
A
B
C
D
1
W
2
X
3
Y
32A
Z
44A
APPLET S/N 32A
CODE KEY C
44A
CRYPTO UNIT UID
DESKTOP PC
22
25
MODEM
24
HARD DRIVE
26
NATIVE SECURITY APPLET
28
ENCRYPTED APPLET 1
30
ENCRYPTED APPLET N
32
32A
CPU
34
ROM
36
TIME CLOCK
38
RAM
40
INPUT/OUTPUT INTERFACE
42
PUBLIC CRYPTOGRAPHIC CONTROL UNIT
44
UID
44A

FIG. 2

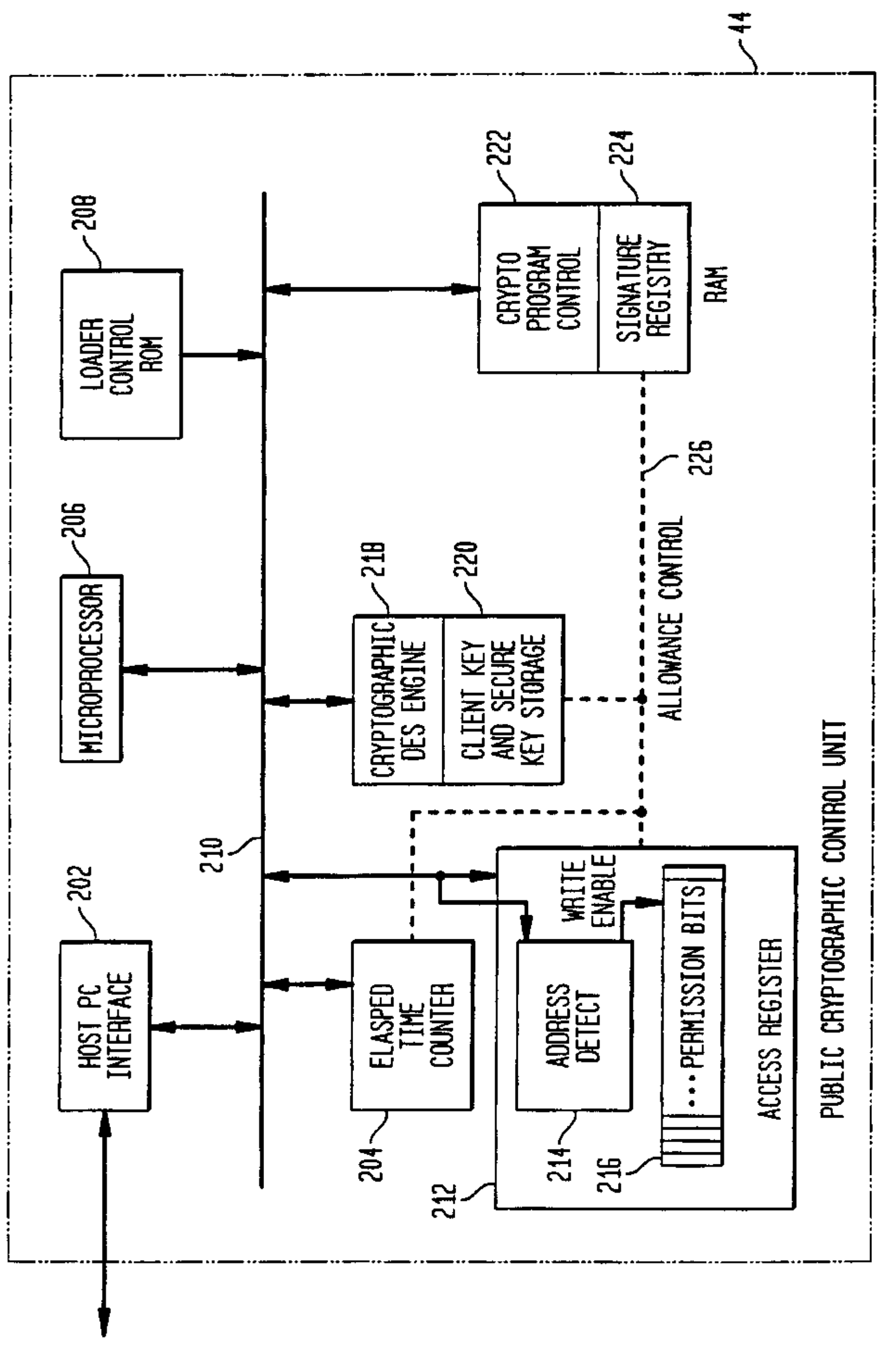

44
HOST PC INTERFACE
202
MICROPROCESSOR
206
LOADER CONTROL ROM
208
210
ELASPED TIME COUNTER
204
CRYPTOGRAPHIC DES ENGINE
218
CLIENT KEY AND SECURE KEY STORAGE
220
CRYPTO PROGRAM CONTROL
222
SIGNATURE REGISTRY
224
RAM
ALLOWANCE CONTROL
226
WRITE ENABLE
ADDRESS DETECT
214
...PERMISSION BITS
216
ACCESS REGISTER
212
PUBLIC CRYPTOGRAPHIC CONTROL UNIT

FIG. 3A

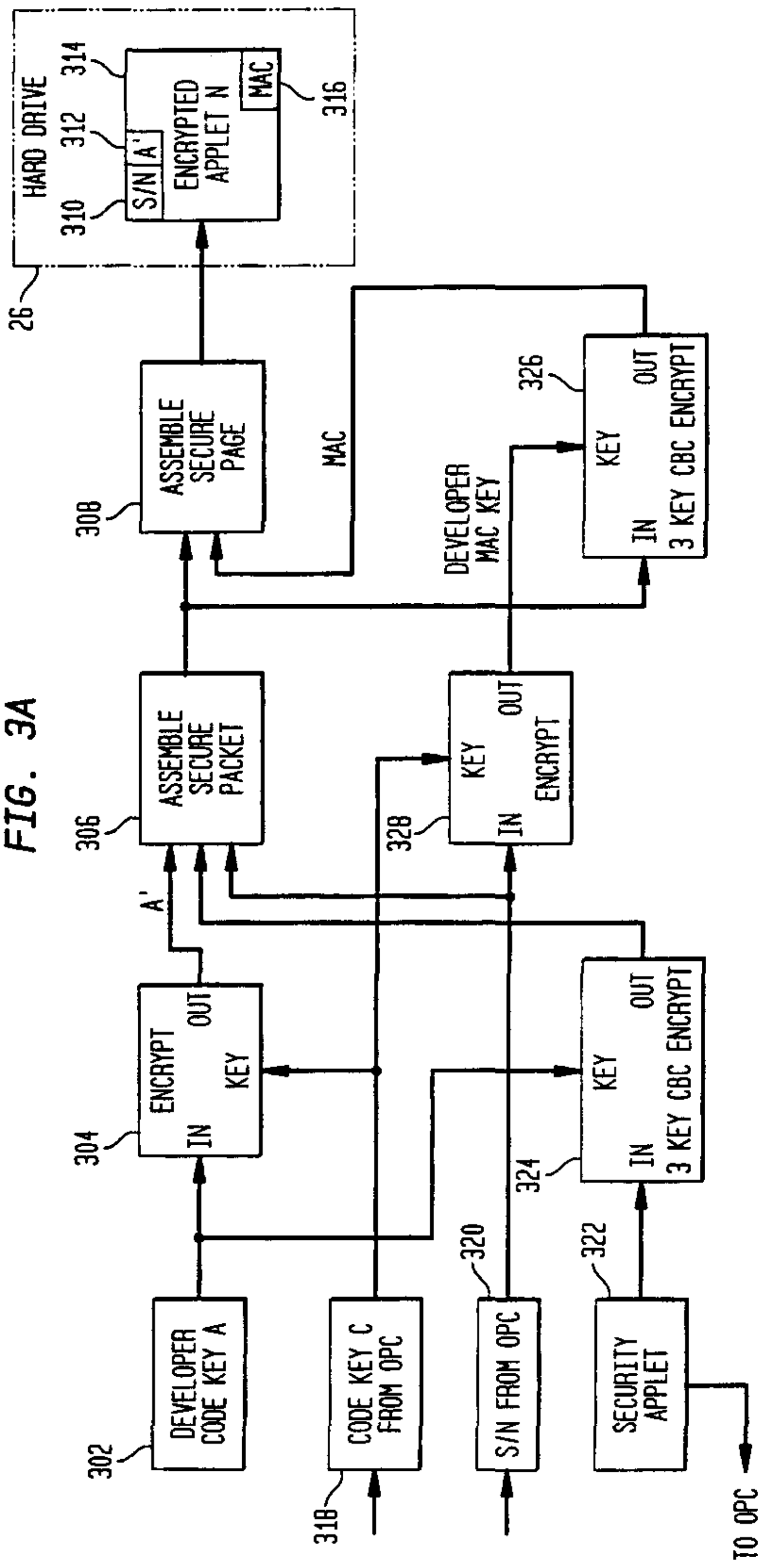

HARD DRIVE
26
310
312
314
S/N
A'
ENCRYPTED APPLET N
MAC
316
ASSEMBLE SECURE PAGE
308
MAC
326
KEY
OUT
IN
3 KEY CBC ENCRYPT
DEVELOPER MAC KEY
ASSEMBLE SECURE PACKET
306
KEY
OUT
IN
ENCRYPT
328
A'
ENCRYPT
IN
KEY
OUT
304
KEY
IN
OUT
3 KEY CBC ENCRYPT
324
DEVELOPER CODE KEY A
302
CODE KEY C FROM OPC
318
S/N FROM OPC
320
SECURITY APPLET
322
TO OPC

FIG. 4

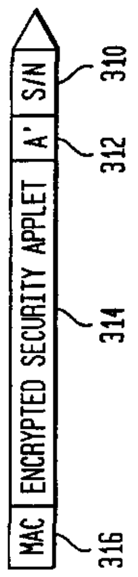

MAC
ENCRYPTED SECURITY APPLET
A'
S/N
316
314
312
310

FIG. 3B

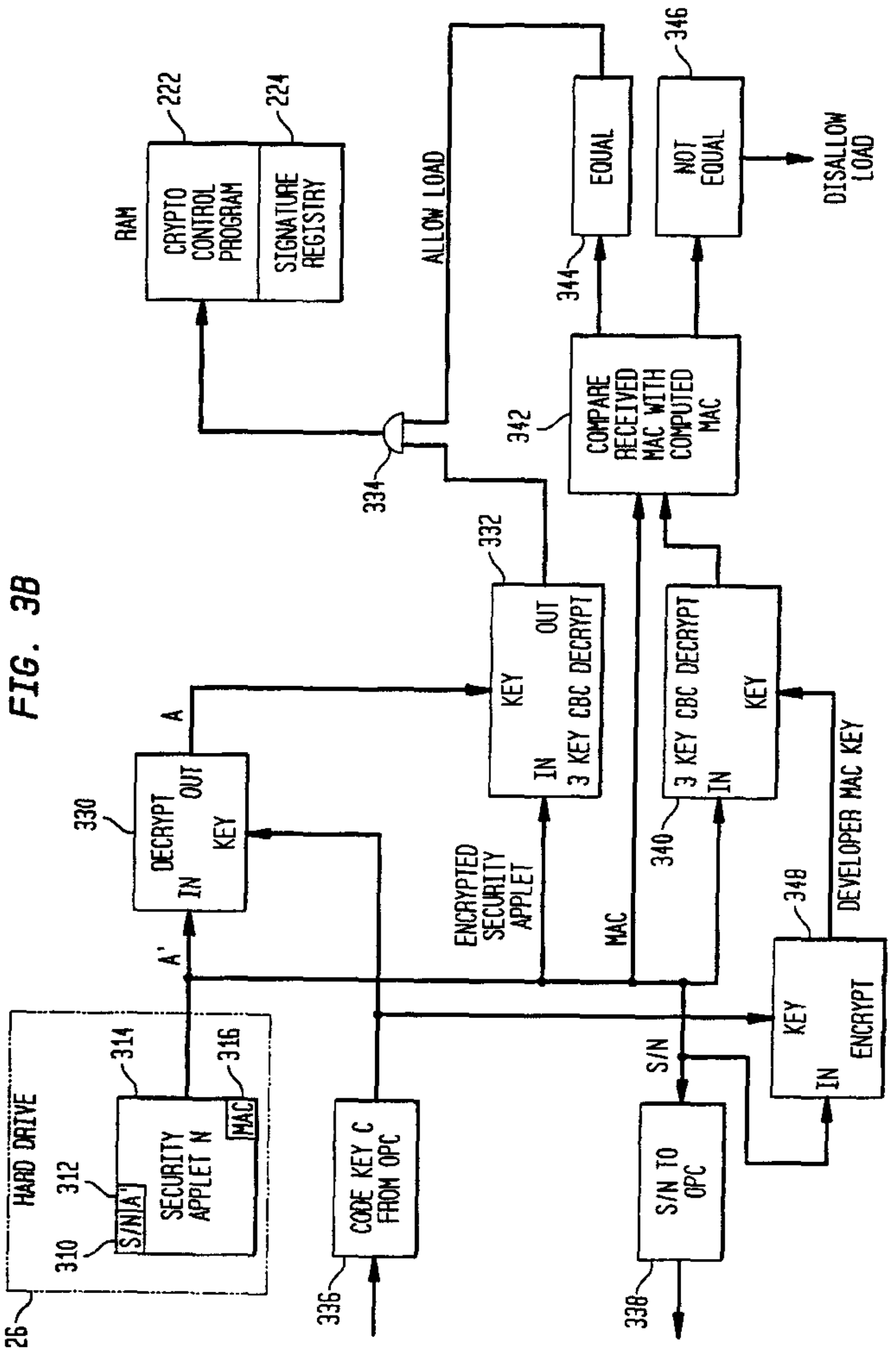

26
HARD DRIVE
310
312
S/N
A'
314
SECURITY APPLET N
316
MAC
336
CODE KEY C FROM OPC
A'
330
DECRYPT
IN
OUT
KEY
A
332
KEY
OUT
IN
3 KEY CBC DECRYPT
ENCRYPTED SECURITY APPLET
MAC
S/N
338
S/N TO OPC
340
3 KEY CBC DECRYPT
IN
KEY
348
KEY
ENCRYPT
IN
DEVELOPER MAC KEY
334
RAM
222
CRYPTO CONTROL PROGRAM
224
SIGNATURE REGISTRY
ALLOW LOAD
342
COMPARE RECEIVED MAC WITH COMPUTED MAC
344
EQUAL
346
NOT EQUAL
DISALLOW LOAD

FIG. 5

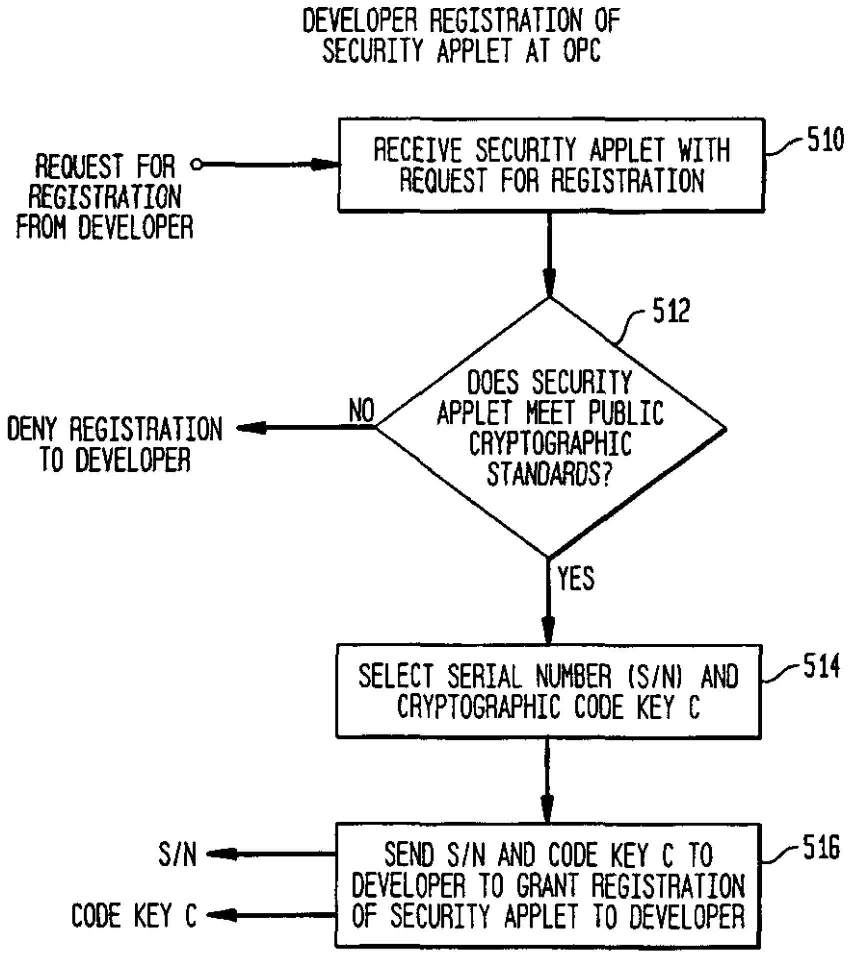

DEVELOPER REGISTRATION OF SECURITY APPLET AT OPC
REQUEST FOR REGISTRATION FROM DEVELOPER
RECEIVE SECURITY APPLET WITH REQUEST FOR REGISTRATION
510
DOES SECURITY APPLET MEET PUBLIC CRYPTOGRAPHIC STANDARDS?
512
NO
DENY REGISTRATION TO DEVELOPER
YES
SELECT SERIAL NUMBER (S/N) AND CRYPTOGRAPHIC CODE KEY C
514
SEND S/N AND CODE KEY C TO DEVELOPER TO GRANT REGISTRATION OF SECURITY APPLET TO DEVELOPER
516
S/N
CODE KEY C

FIG. 6

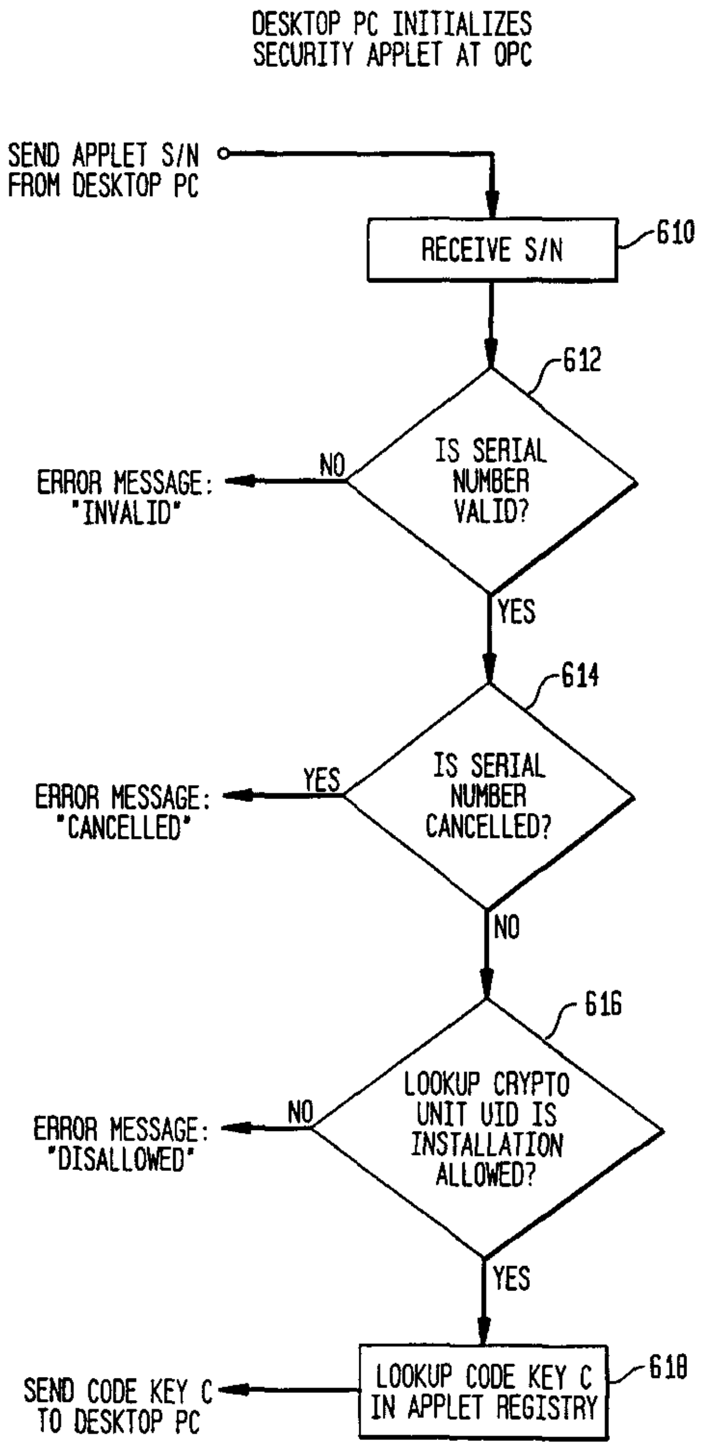

DESKTOP PC INITIALIZES SECURITY APPLET AT OPC
SEND APPLET S/N FROM DESKTOP PC
RECEIVE S/N
610
612
IS SERIAL NUMBER VALID?
NO
ERROR MESSAGE: "INVALID"
YES
614
IS SERIAL NUMBER CANCELLED?
YES
ERROR MESSAGE: "CANCELLED"
NO
616
LOOKUP CRYPTO UNIT UID IS INSTALLATION ALLOWED?
NO
ERROR MESSAGE: "DISALLOWED"
YES
LOOKUP CODE KEY C IN APPLET REGISTRY
618
SEND CODE KEY C TO DESKTOP PC

FIG. 7

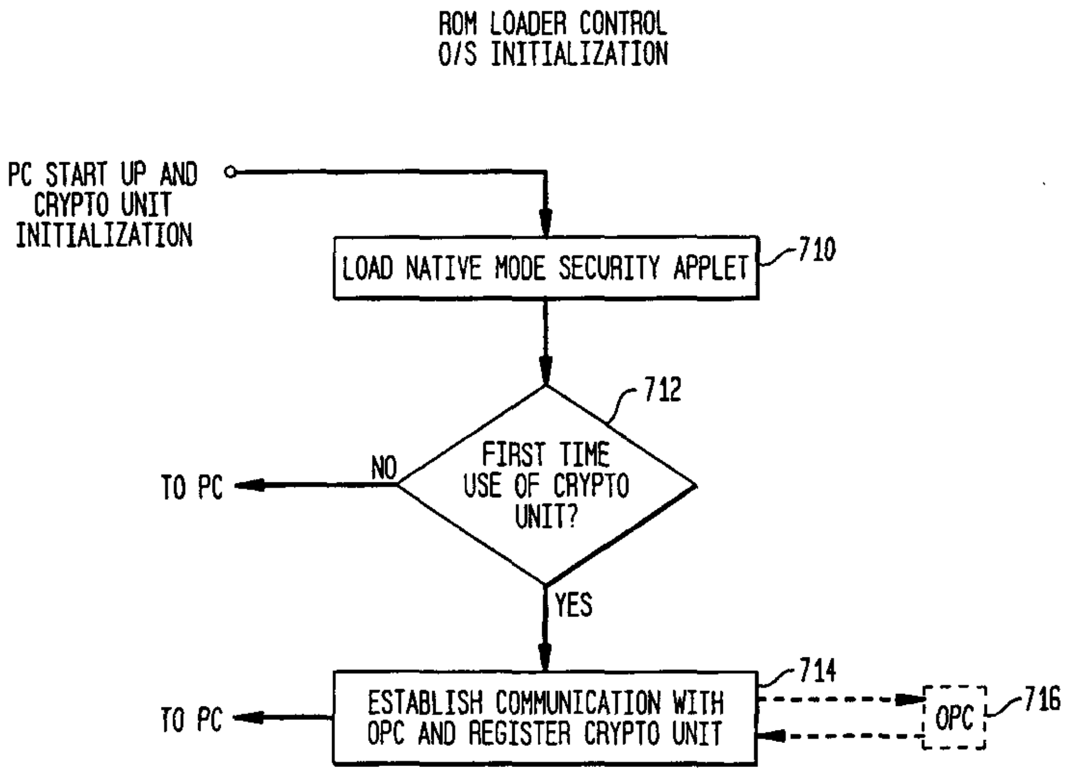

ROM LOADER CONTROL
O/S INITIALIZATION
PC START UP AND CRYPTO UNIT INITIALIZATION
LOAD NATIVE MODE SECURITY APPLET
710
712
FIRST TIME USE OF CRYPTO UNIT?
NO
TO PC
YES
714
ESTABLISH COMMUNICATION WITH OPC AND REGISTER CRYPTO UNIT
TO PC
OPC
716

FIG. 8A

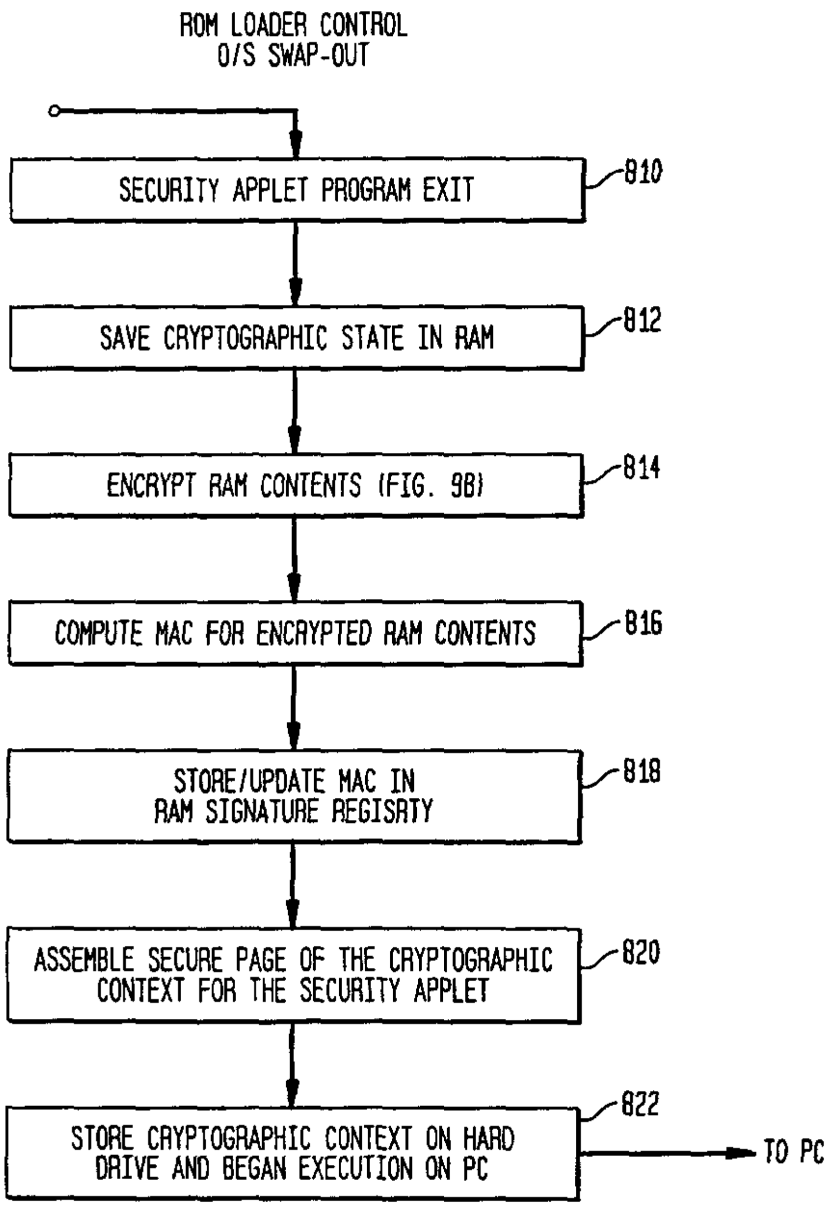

ROM LOADER CONTROL
O/S SWAP-OUT
SECURITY APPLET PROGRAM EXIT
810
SAVE CRYPTOGRAPHIC STATE IN RAM
812
ENCRYPT RAM CONTENTS (FIG. 9B)
814
COMPUTE MAC FOR ENCRYPTED RAM CONTENTS
816
STORE/UPDATE MAC IN RAM SIGNATURE REGISRTY
818
ASSEMBLE SECURE PAGE OF THE CRYPTOGRAPHIC CONTEXT FOR THE SECURITY APPLET
820
STORE CRYPTOGRAPHIC CONTEXT ON HARD DRIVE AND BEGAN EXECUTION ON PC
822
TO PC

FIG. 8B

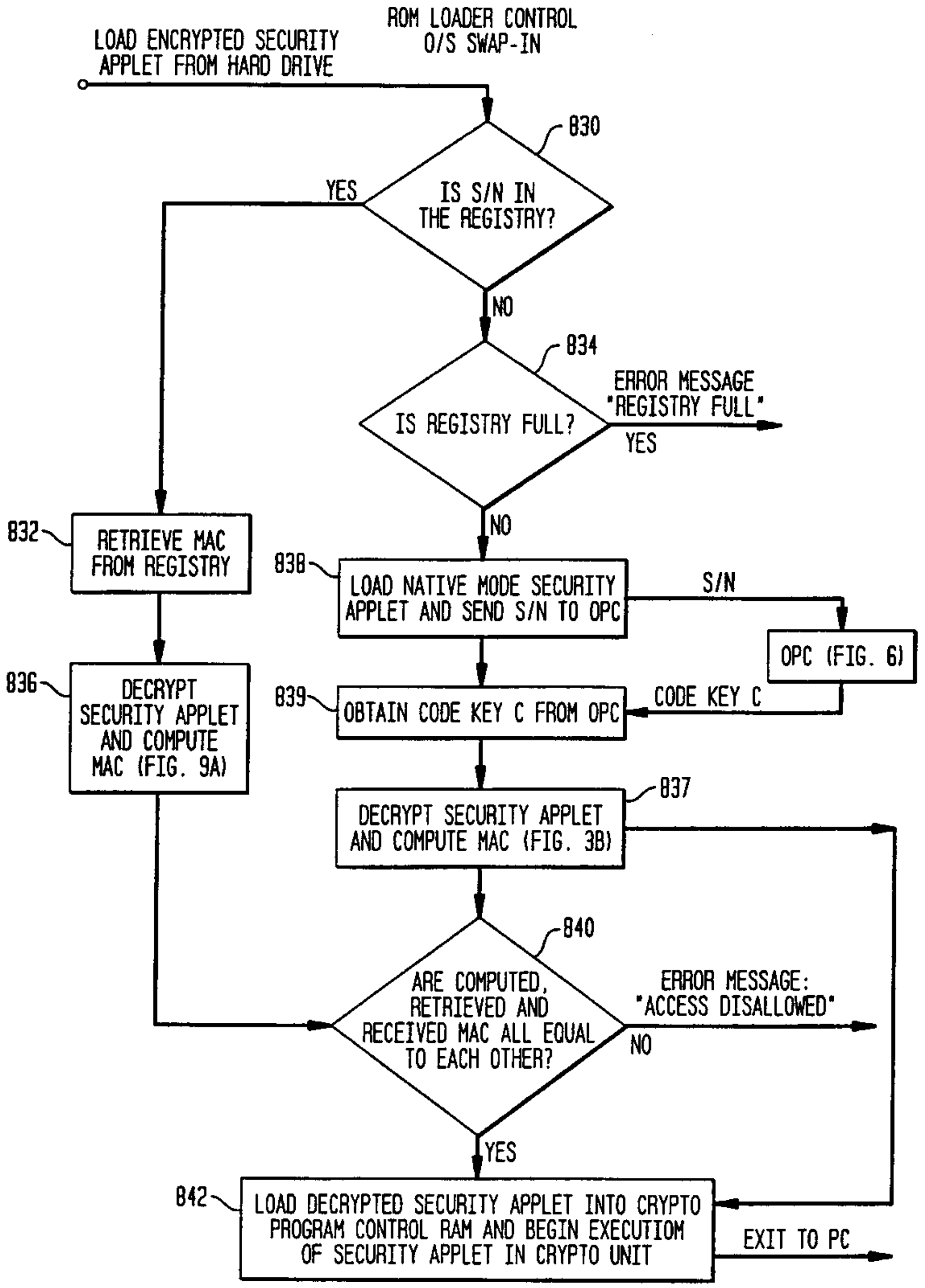
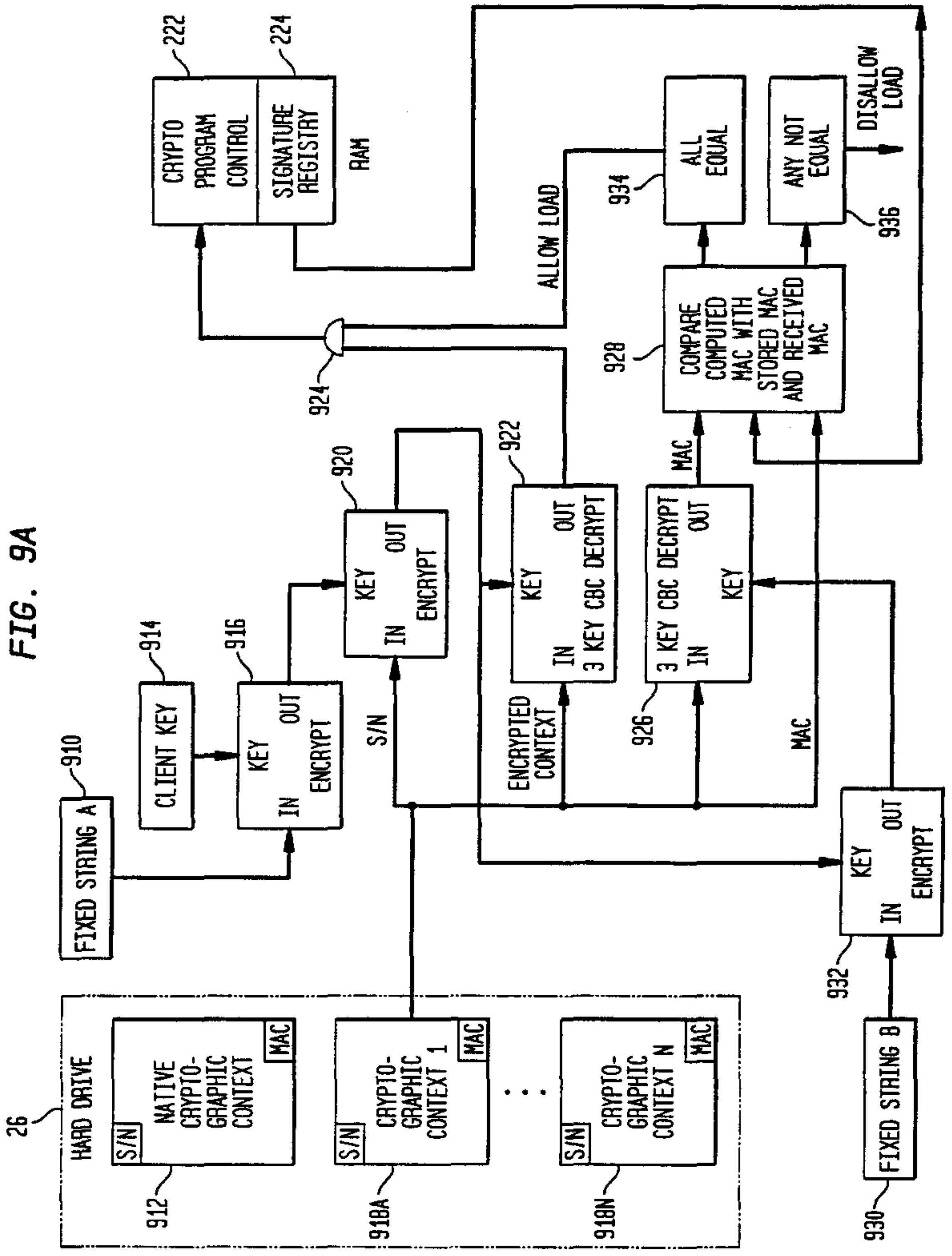
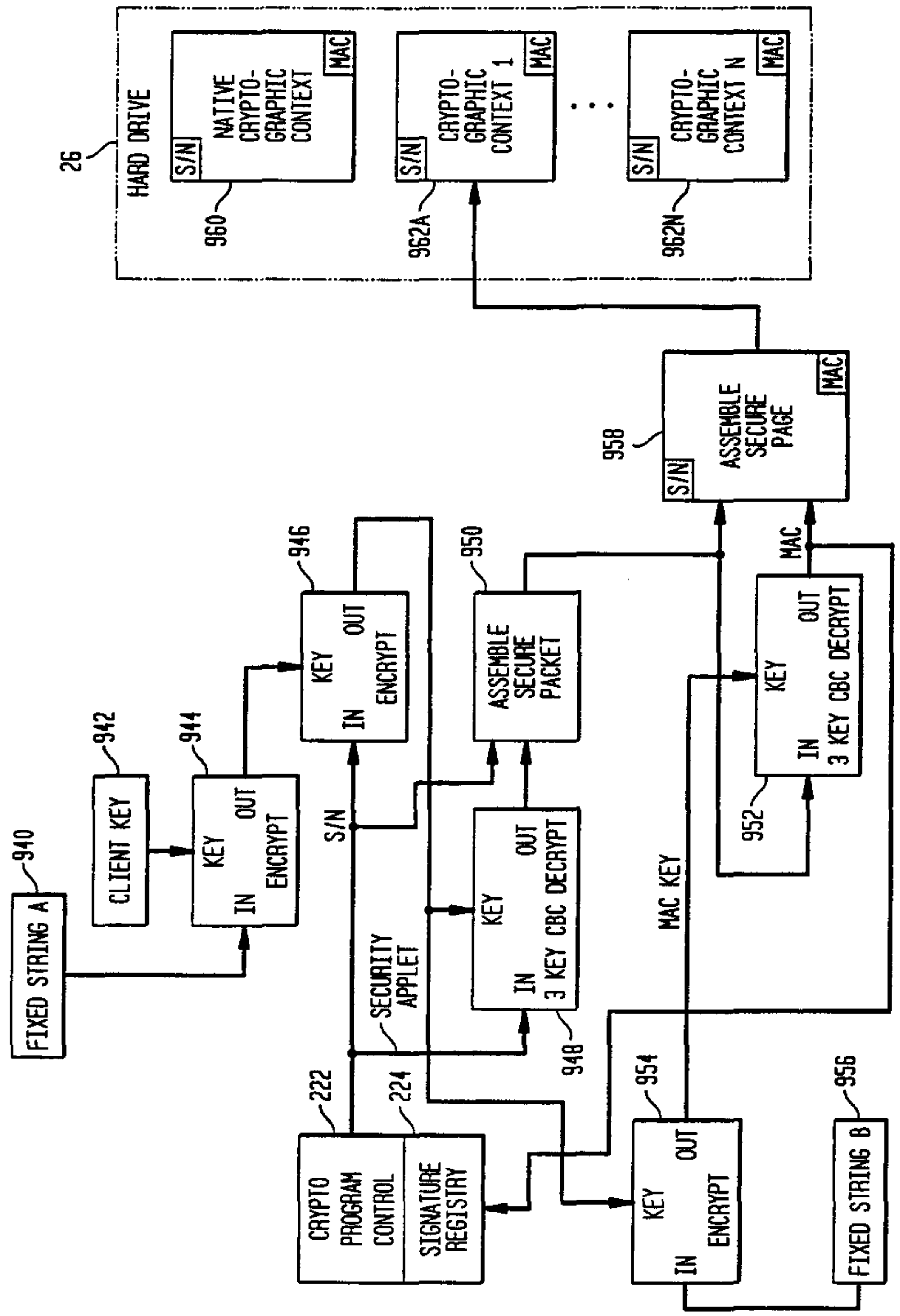

ROM LOADER CONTROL
O/S SWAP-IN
LOAD ENCRYPTED SECURITY APPLET FROM HARD DRIVE
830
IS S/N IN THE REGISTRY?
YES
NO
834
IS REGISTRY FULL?
ERROR MESSAGE "REGISTRY FULL"
YES
NO
832
RETRIEVE MAC FROM REGISTRY
838
LOAD NATIVE MODE SECURITY APPLET AND SEND S/N TO OPC
S/N
OPC (FIG. 6)
836
DECRYPT SECURITY APPLET AND COMPUTE MAC (FIG. 9A)
839
OBTAIN CODE KEY C FROM OPC
CODE KEY C
837
DECRYPT SECURITY APPLET AND COMPUTE MAC (FIG. 3B)
840
ARE COMPUTED, RETRIEVED AND RECEIVED MAC ALL EQUAL TO EACH OTHER?
ERROR MESSAGE: "ACCESS DISALLOWED"
NO
YES
842
LOAD DECRYPTED SECURITY APPLET INTO CRYPTO PROGRAM CONTROL RAM AND BEGIN EXECUTIOM OF SECURITY APPLET IN CRYPTO UNIT
EXIT TO PC

FIG. 9A



26
HARD DRIVE
S/N
NATIVE CRYPTO-GRAPHIC CONTEXT
MAC
912
S/N
CRYPTO-GRAPHIC CONTEXT 1
MAC
918A
S/N
CRYPTO-GRAPHIC CONTEXT N
MAC
918N
FIXED STRING A
910
CLIENT KEY
914
916
KEY
IN
OUT
ENCRYPT
920
KEY
IN
OUT
ENCRYPT
S/N
922
KEY
IN
OUT
3 KEY CBC DECRYPT
ENCRYPTED CONTEXT
926
3 KEY CBC DECRYPT
IN
OUT
KEY
MAC
MAC
928
COMPARE COMPUTED MAC WITH STORED MAC AND RECEIVED MAC
924
ALLOW LOAD
934
ALL EQUAL
936
ANY NOT EQUAL
DISALLOW LOAD
222
CRYPTO PROGRAM CONTROL
224
SIGNATURE REGISTRY
RAM
932
KEY
IN
OUT
ENCRYPT
FIXED STRING B
930

FIG. 9B


26
HARD DRIVE
S/N
NATIVE CRYPTO-GRAPHIC CONTEXT
MAC
960
S/N
CRYPTO-GRAPHIC CONTEXT 1
MAC
962A
S/N
CRYPTO-GRAPHIC CONTEXT N
MAC
962N
958
S/N
ASSEMBLE SECURE PAGE
MAC
940
FIXED STRING A
942
CLIENT KEY
944
KEY
OUT
IN
ENCRYPT
946
KEY
OUT
IN
ENCRYPT
950
ASSEMBLE SECURE PACKET
MAC
KEY
OUT
IN
3 KEY CBC DECRYPT
952
S/N
KEY
OUT
IN
3 KEY CBC DECRYPT
948
SECURITY APPLET
MAC KEY
222
224
CRYPTO PROGRAM CONTROL
SIGNATURE REGISTRY
954
KEY
OUT
IN
ENCRYPT
956
FIXED STRING B

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 00/13239

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5 764 762 A (KAZMIERCZAK GREGORY J ET AL) 9 June 1998 (1998-06-09) abstract; figure 1 column 2, line 40 -column 3, line 30 column 4, line 25 -column 6, line 30 column 11, line 63 -column 14, line 50 ---	1-6, 9-14, 17-22
Y	EP 0 833 241 A (MITSUBISHI CORP) 1 April 1998 (1998-04-01) abstract; figures 3A-3D column 4, line 26 -column 5, line 11 column 11, line 54 -column 16, line 15 --- -/--	1-6, 9-14, 17-22

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search 8 August 2000	Date of mailing of the international search report 16/08/2000
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Authorized officer Sigolo, A

2

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Inte ;onal Application No
PCT/US 00/13239

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0 555 715 A (IBM) 18 August 1993 (1993-08-18) abstract; figure 3 page 3, line 51 -page 4, line 19 page 5, line 15 -page 6, line 35 -----	1-22

2

INTERNATIONAL SEARCH REPORT

Information on patent family members

Inte onal Application No
PCT/US 00/13239

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5764762 A	09-06-1998	US 5615264 A	25-03-1997
		AU 6635396 A	09-01-1997
		EP 0836774 A	22-04-1998
		WO 9642153 A	27-12-1996
EP 0833241 A	01-04-1998	JP 10107787 A	24-04-1998
EP 0555715 A	18-08-1993	US 5301231 A	05-04-1994
		JP 5334073 A	17-12-1993

[19]中华人民共和国国家知识产权局

[51]Int. Cl^7
G06F 1/00

[12]发明专利申请公开说明书

[21]申请号 00807632.4

[43]公开日 2002年5月22日

[11]公开号 CN 1350670A

[22]申请日 2000.5.15 [21]申请号 00807632.4
[30]优先权
[32]1999.5.17 [33]US [31]09/313,295
[86]国际申请 PCT/US00/13239 2000.5.15
[87]国际公布 WO00/70429 英 2000.11.23
[85]进入国家阶段日期 2001.11.16
[71]申请人 波系统股份有限公司
地址 美国马萨诸塞州
[72]发明人 S·K·斯普拉格
G·J·卡滋梅尔克扎克

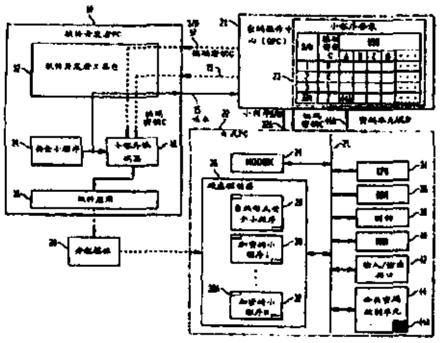
[74]专利代理机构 上海专利商标事务所
代理人 李 湘

权利要求书6页 说明书16页 附图页数11页

[54]发明名称 公共密钥控制单元及其系统

[57]摘要

一个普遍适用的公共密码控制单元(密码单元)被用于被多个独立用户共享 的密码系统。作外围设备安装在通用计算机的密码单元加载和卸载加密的安全小 程序到运行每个安全小程序的密码单元的 RAM 存储器。密码单元和包括它的系统 提供一个安全的内部环境,其中只有预先批准的安全小程序被授予加载和运行的 许可。在密码单元中的计算环境由于与每个密码单元通讯的密码操作中心(OPC) 而成为安全的。在分配给定的安全小程序以便得到对给定安全小程序的必要的许 可以前,软件开发者提交一个提出的安全小程序给 OPC。仅当从 OPC 得到所有必 要的许可时,一个给定的安全小程序才被允许去密码单元内加载运行。当第一安 全小程序结束运行时,密码将加密形式的当前加载的第一安全小程序卸载(换上) 到 PC 的硬盘,并加载(换入)下一个安全小程序。每个安全小程序的密码内容 被保存储存在 PC 硬盘驱动器的文件中。以那样的方式,单个密码单元在多个独 立的用户之间被共享。



ISSN1008-4274

知识产权出版社出版

权　利　要　求　书

1、一种用于密码密钥分配系统的方法，系统包括一个具有一个密码控制单元的用户计算机、一个软件开发者计算机和一个密码操作中心，其特征在于所述方法包括：

在所述的软件开发者计算机上产生另一个安全小程序；

将所述的第一程序从所述软件开发者计算机发送到所述密码操作中心；

在所述的软件开发者计算机上从所述的密码操作中心接收第一密码密钥；

在所述软件开发者计算机上从所述密码操作中心接收第一序列号；

在加密所述第一安全小程序以形成第一加密的安全小程序的过程中使用所述的第一密码密钥；

将所述第一序列号附列至所述第一加密安全小程序上以形式第一安全包；以及

分配所述第一安全包到所述用户计算机。

2、如权利要求 1 所述的方法，其特征在于所述密码控制单元包括一个程序控制存储器，所述方法还包括：

将所述第一系列号从所述密码控制单元发送到所述密码操作中心；

在所述密码控制单元接收来自所述密码操作中心的第一密码密钥；

在从所述第一加密安全小程序解密所述第一安全小程序的过程中使用第一密码密钥；以及

将所述第一安全小程序加载到所述程序控制存储器。

3、如权利要求 1 所述的方法，其特征在于在加密所述第一安全小程序以形成一加密的安全小程序的过程中使用所述第一密码密钥的所述步骤还包括：

在所述软件开发者计算机上借助于程序员密钥加密所述第一安全小程序以形成程序员加密的安全小程序；

在所述软件开发者计算机上借助于所述第一密码密钥加密所述程序员密钥以形成第一加密的程序员密钥；

附上所述第一加密的程序员密钥和所述第一程序员加密的安全小程序以形成所述第一加密的安全小程序。

4、如权利要求 2 所述的方法，其特征在于所述软件开发者计算机还包括借助所述第一密码密钥加密的程序员密钥以形成在所述第一加密的安全小程序中的第一加密的程序员密钥；而且当在所述用户计算机上从所述第一加密安全小程序解密所述第一安全小程序的过程中使用所述第一密码密钥的所述步骤包括：

在所述用户计算机上接收包括所述第一程序员加密的安全小程序的所述第一

安全包；

在所述用户计算机借助于所述第一密码密钥解密所述第一加密的程序员密钥以形成一个复原的程序员密钥；以及

借助所述复原的程序员密钥解密所述第一程序员加密安全小程序以复原所述的第一安全小程序。

5、如权利要求 1 所述的方法，其特征在于还包括：

存储一个记录多个安全小程序、密码密钥和序列号之间对应关系的安全小程序登录表，所述安全小程序登录表具有一个条目，指出所述第一序列号和所述第一密码密钥对应于所述第一安全小程序。

6、如权利要求 2 所述的方法，其特征在于所述用户计算机包括第一用户识别号，所述方法还包括：

从所述用户计算机将所述第一用户识别号发送到所述密码操作中心；

存储一个记录多个安全小程序、密码密钥、序列号和用户识别号之间对应关系的安全小程序登录表，所述安全小程序登录表具有一个条目，指出所述第一序列号，所述第一密码密钥和所述第一用户识别号对应于所述第一安全小程序。

7、如权利要求 2 所述的方法，其特征在于所述用户计算机还包括一个用户计算机硬盘驱动存储器，且所述系统还包括第二软件开发者计算机、一个具有相应第二序列号及与之对应的第二密码密钥的第二安全小程序，所述方法还包括：

在使用第一用户计算机密钥形成第一加密的安全内容的过程中加密所述用户控制存储器的内容；

将所述第一加密的安全内容储存在所述用户计算机硬盘驱动存储器；且在所述程序控制存储器加载所述第二安全小程序。

8、如权利要求 7 所述的方法，其特征在于还包括：

在使用第二用户计算机密钥形成第二加密安全内容的过程中加密所述用户控制存储器的内容；

将所述第二加密的安全内容存储在所述用户计算机的硬盘驱动器；

在使用所述第一用户计算机密钥复原所述第一安全内容的过程中解密所述第一加密安全内容；且

在所述程序控制存储器加载所述第一安全内容。

9、一种用于密码密钥分配系统的装置，系统包括一个具有一个密码控制单元的用户计算机、一个软件开发者计算机和一个密码操作中心，其特征在于所述装置包括：

用于在所述软件开发者计算机上产生第一安全小程序的装置；

用于将所述第一安全小程序从所述软件开发者计算机发送到所述密码操作中

心的装置；

用于在所述软件开发者计算机从所述密码操作中心接收第一密码密钥的装置；

用于在所述软件开发者计算机从所述密码操作中心接收第一序列号的装置；

用于在解密所述第一安全小程序以形成第一解密安全小程序的过程中使用的第一密码密钥的装置；

用于将所述第一序列号附在所述第一解密安全小程序以形成第一安全包的装置；和

用于将所述第一安全包分配到所述用户计算机的装置。

10、如权利要求 9 所述的装置，其特征在于所述密码控制单元包括一个程序控制存储器，所述装置还包括：

用于将所述第一序列号从所述密码控制单元发送到所述密码操作中心的装置；

用于在所述密码控制单元上从所述密码操作中心接收所述第一密码密钥的装置；

用于在从所述第一加密的安全小程序解密所述第一安全小程序的过程中使用所述密码密钥的装置；和

用于将所述第一安全小程序加载到所述程序控制存储器的装置。

11、如权利要求 9 所述的装置，其特征在于用于在加密所述第一安全小程序以形成第一加密安全小程序的过程中使用所述第一密码密钥的装置还包括：

用于在所述软件开发者计算机上借助一个程序员密钥加密所述第一安全小程序以形成第一程序员加密的安全小程序的装置；

用于在所述软件开发者计算机上借助所述第一密码密钥加密所述程序员密钥以形成第一加密的程序员密钥的装置；

用于附上所述第一加密的程序员密钥和所述第一程序员加密安全小程序以形成所述第一加密的安全小程序的装置。

12 如权利要求 10 所述的装置，其特征在于所述软件开发者计算机还包括一个借助于所述第一密码密钥加密的程序员密钥，它用于形成在所述第一加密安全小程序中的第一加密程序员密钥，而且用于所述用户计算机上从所述第一程序员加密的安全小程序解密所述第一安全小程序的过程中使用第一密码密钥的所述装置包括：

用于在所述用户计算接收包括所述第一程序员加密的安全小程序的所述第一安全包的装置；

用于在所述用户计算机上借助所述第一密码密钥解密所述第一加密程序员密

钥以形成一个复原的程序员密钥的装置；

用于借助所述复原的程序员密钥解密所述第一程序员加密的安全小程序以复原所述第一安全小程序的装置。

13、如权利要求 9 所述的装置，其特征在于还包括：

用于存储记录多个安全小程序、密码密钥和序列号之间对应关系的安全小程序登录表的装置，所述安全小程序登录表具有一个条目，指出第一序列号和所述第一密码密钥对应于所述第一安全小程序。

14、如权利要求 10 所述的装置，其特征在于所述用户计算机包括一个第一用户识别号，所述装置还包括：

用于将所述第一用户识别号从所述用户计算机发送到所述密码操作中心；

用于存储记录多个安全小程序、密码密钥、序列号和用户识别号之间对应关系的一个安全小程序登录表的装置，所述安全小程序登录表具有一个条目，指出所述第一序列号、所述第一密码密钥和所述第一用户识别号对应于所述第一安全小程序。

15、如权利要求 10 所述的装置，其特征在于所述用户计算机还包括一个用户计算机硬盘驱动存储器，且所述系统还包括一个第二软件开发计算机、一个具有相应的第二序列号及与之对应的第二密码密钥的第二安全小程序，所述装置还包括：

用于在使用第一用户计算机形成第一加密的安全内容的过程中加密所述用户控制存储器的装置；

用于将第一加密的安全内容存入所述用户计算机的硬盘驱动存储器装置；和

用于将所述第二安全小程序加载到的程序控制存储器的装置。

16、如权利要求 15 所述的装置，其特征在于还包括；

用于在使用第二用户计算机密钥形成第二加密安全内容的过程中加密所述用户控制存储器的装置；

用于将所述第二加密的安全内容存入所述用户计算机硬盘驱动存储器的装置；

用于在使用所述第一用户计算机密钥所述第一安全小程序的过程中解密所述第一加密的安全内容的装置；和

用于将所述第一安全内容加载到所述程序控制存储器的装置。

17、一种用于密码密钥分配系统的方法，系统包括一个具有一个密码控制单元的用户计算机、一个软件开发者计算机和一个密码操作中心，所述软件开发者计算机产生第一安全小程序，它在使用第一密码密钥形成第一加密的安全小程序的过程中加密所述的第一安全小程序，并将所述第一加密的安全小程序分配到所

述的用户计算机，其特征在于在所述密码操作中心上的密码密钥分配方法包括：

在所述密码操作中心从所述软件开发者计算机接收所述第一安全程序；

将第一序列号从所述密码操作中心发送到所述软件开发者计算机；

将第一密码密钥从所述密码操作中心发送到所述软件开发者计算机；

在所述密码操作中心从所述密码控制单元接收所述第一序列号；

将所述第一密码密钥从所述密码操作中心发送到所述密码控制单元。

18、如权利要求 17 所述的方法，其特征在于所述密码控制单元包括一个程序控制存储器，所述的方法还包括：

在所述密码控制单元接收包括第一序列号的第--加密安全小程序；

将所述第一序列号从所述密码控制单元发送到所述密码操作中心；

在所述密码控制单元从所述密码操作中心接收所述第一密码密钥；

在从所述第一加密的安全小程序解密所述第一安全小程序的过程中使用所述第一密码密钥；和

将所述第一安全小程序加载到所述程序控制存储器。

19、一种用于密码密钥分配系统的方法，系统具有一个软件开发者计算机和一个密码操作中信，所述软件开发者计算机产生一个由第一序列号识别的第一安全小程序并在使用第一密码密钥形成第一加密安全小程序的过程中加密所述第一安全小程序，所述系统进一步包括具有带一个程序控制存储器的密码控制单元的用户计算机，其特征在于所述方法包括：

在所述密码控制中心接收包括所述第一序列号的第一安全小程序；

将所述第一序列号发送到所述密码操作中心；

在所述密码控制单元从所述密码操作中心接收所述第一密码密钥；

在从所述第一加密安全小程序解密所述第一安全小程序的过程中使用所述第一密码密钥；和

将所述第一安全小程序加载到所述程序控制存储器。

20、在一个密码密钥分配系统中，包括一个具有密码控制单元的用户计算机、一个软件开发者计算机和一个密码操作中心，所述软件开发者计算机产生一个第一安全小程序并在使用第一密码密钥形成第一加密的安全小程序的过程中加密所述第一安全小程序，并将所述第一安全小程序分配到所述用户计算机，其特征在于所述密码操作中心处的密码密钥分配装置包括：

用于在所述密码操作中心从所述软件开发者计算机接收所述第一安全小程序的装置；

用于将第一序列号从所述密码操作中心发送到所述软件开发者计算机的装置；

用于将所述第一密码密钥从所述密码操作中心发送到所述软件开发者计算机的装置；

用于在所述密码操作中心从所述密码控制单元接收所述第一序列号的装置；

用于将所述第一密码密钥从所述密码操作中心发送到所述密码控制中心的装置。

21、如权利要求 20 所述的装置，其特征在于所述密码控制单元包括一个程序控制存储器，所述装置还包括：

用于在所述密码控制单元接收包括所述第一序列号的第一加密的安全小程序的装置；

用于将所述第一序列号从所述密码控制单元发送到所述密码操作中心的装置；

用于在所述密码控制单元从所述密码操作中心接收所述第一密码密钥的装置；

用于在从所述第一安全小程序解密所述第一安全小程序的过程中使用所述第一密码密钥的装置；和

用于将所述第一安全小程序加载到所述程序控制存储器的装置。

22、在一个密码密钥分配系统中，具有一个软件开发者计算机和一个密码操作中心，所述软件开发者计算机产生一个由第一序列号识别的第一安全小程序并在使用第一密码密钥形成第一加密安全小程序的过程中加密所述第一安全小程序，所述系统还包括具有带程序控制存储器的密码控制单元的用户计算机，其特征在于装置包括：

用于在所述密码控制单元接收包括第一序列号的第一安全小程序的装置；

用于将第一序列号发送到所述密码操作中心的装置；

用于在所述密码控制单元从所述密码操作中心接收所述第一密码密钥的装置；

用于在从所述第一加密的安全小程序解密所述第一安全小程序的过程中使用所述第一密码密钥的装置；

将所述第一安全小程序加载到所述程序控制存储器。

说　　明　　书

公共密钥控制单元及其系统

发明领域

本发明涉及密码系统，尤其是涉及一个密钥管理系统及一个共享的公共密码控制单元。

发明背景

许多计算机应用需要实现一个或多个安全功能。计算机程序的安全功能是它防止用户胡乱篡改的特点及操作。

例如，一个软件程序可以具有一个过时日期，在此日期之后程序变成不能运行。但是典型的软件过时功能是不安全的，因为只要将当地计算机时间设置成较早的时间，或者修改该软件使其跳过核时当地计算机时钟的软件，此功能就失效了。

作为另一个例子，一个保存从当地的加密数据度存放数据的记录用于支付当地加密数据库的登记使用的计算机程序通常具有两个关键的寄存器。第一个寄存器表示过去数据使用的量，另一个寄存器表示余数的量。如果更新该寄存器和存款寄存器不是一个安全的功能，用户能减少使用寄存器的内容和／或增加存款寄存器的内容使该系统失效。类似地，为租赁支付目的保存租赁软件的使用记录的租赁软件也需要防止用户对租赁记账寄存器和其他重要的内部寄存器及功能的胡乱篡改。

作为另一个例子，一个远程访问的数据库能对授权的用地该数据库的访问收费。常需要一个安全系统在授权访问该数据库之前识别每个用户的身份。然而另一个安全系统是密钥管理系统，即将密码密钥分配给授权的用户。

一类安全功能解决方法是在软件中实现安全功能。在软件中实现安全功能具有经济上的优点。软件实现也具有通用的优点。但是，在软件中实现软件安全性功能不象在硬件中实现安全功能那样安全。另一方面，与软件相比，硬件实现安全功能费用更高，且可能对每个应用需要专用的硬件。如果每个应用需要它本身的专用的硬件，硬件实现安全功能将是不通用的。

发明内容

本发明以一个方法和装置实现，使用密码控制单元作为在由多个独立的用户共享的系统中广泛适用的公共密码控制单元（密码单元）。

密码单元包括一个具有允许安全共享密码单元资源的专用硬件和固件的通用计算机处理器。尤其是该密码单元包括一个带有只读存储器（ROM）控制程序的专用核心的微处理器芯片，一个通用随机存储器（RAM），一个实时时钟和一个主机输入／输出接口（即与台式 PC 的连接）。此外，该密码单元包括一个 DES（数据加密标准）引擎，用于密码密钥的安全非易失性存储，一个签名寄存 RAM 存储器和专用访问寄存器。

密码单元作为外围设备安装在任何如台式 PC 那样的通用计算机中。使单元成为“公共”的密码控制单元是它作一个安全的计算资源适用于在 PC 上运行的主要应用程序。

为了使用密码单元资源，对应于安全功能的主应用程序的一个部分被储存在 PC 上。这里称为安全小程序（Secure Applet）的安全功能加载到运行每个安全小程序的密码单元的板上的 RAM 或从上卸载。模仿下载并运行内部浏览器的 Jave 小程序（Jave applet），安全小程序是可移植的、可执行的文件以便加载到适合的计算实体并实现一个或多个安全功能。在此意义上，此密码单元象一个专用的协处理器，适用于运行安全应用程序（小程序）。

PC 使安全小程序加载到密码单元的程序控制单元，后者运行该小程序并将安全功能的结果返回到 PC。但是，不象一个通常的协处理器，对密码单元的访问不完全在台式 PC 的控制下。即，台式 PC 可以不加载和运行任何安全小程序。密码单元和系统的一部分提供其安全内部环境，只有某些安全小程序被许可加载并在密码单元运行。

为保障在密码单元的计算环境，提供一个密码运行中心（OPC），它与密码单元通讯。尤其是首次遇到新安全小程序，并在该新安全小程序被允许在公共密码单元运行以前，该密码单元与 OPC 通讯。在新密码单元首次被安装在台式 PC 时该密码单元也与 OPC 通讯。该密码单元也在定期时间周期基础上与 OPC 通讯。此外，软件开发者在为了得到必要的许可，使给定的安全小程序加载并运行在该密码单元而分配给定的安全小程序以前也要与 OPC 通讯。仅当从 OPC 获得所有必要的许可时，给定的安全小程序被允许加载并运行在密码单元。

操作系统

密码单元操作系统（O/S）包括两个部分：一个 ROM 加载控制程序，和一个自然形式安全小程序（native mode security applet），ROM 加载控制程序是控制程序的压缩专用核心，它被储存在密码单元的 ROM 中。可以由软盘、CDROM 或电话线调制解调器分配的自然形式安全小程序是可移植及可写的文件，通常被存入台式 PC 的硬盘中。

重要的安全功能在 ROM 加载控制程中实现。尤其是，ROM 加载控制程序控制将安全小程序到密码单元及外部资源的加载和从其中卸载，包括将自然形式安全小程序的加载和卸载。

自然模式安全小程序具有两个主要功能：在首次使用密码单元时将密码单元登录到 OPC，并授予首次使用每个单元应用安全小程序的许可。作为一般规则，每当该密码单元与 OPC 通讯时使用自然形式安全小程序。

系统操作

在其安全软件应用程序中希望使用公共密码控制单元的应用软件开发者首先提交一份提出的安全小程序给 OPC 作为考虑。该提出的安全小程序必须满足包括安全标准的某些标准。例如，提出的安全小程序必须小到足以适合到在密码单元的板上 RAM。OPC 进一步审查该提出的安全小程序是否符合安全标准。

在完成所有安全符合性测试以后，OPC 授予或拒绝该提出的安全小程序使用庐密码的许可。允许使用所提出的安全小程序包括指定一个序列号及编码密钥 C 给该提出的安全小程序。序列号及编码密钥 C 被存入 OPC 中的小程序登录处。开发者在一个加密该建议的安全小程序的过程中使用庐编码密钥 C，并使用该序列号识别该加密的安全小程序。

在启动台式 PC 初始化时，密码 ROM 加载控制程序将自然形式安全小程序加载到在该密码单元的板上 RAM 中。虽然 ROM 加载控制程序以前已经从 OPC 授予加载到并在该密码单元运行的许可，它还测试该自然形式安全小程序。

ROM 加载控制程序方便了在多个用户之中共享使用该密码单元。尤其是，ROM 加载控制程序将自然形式安全小程序从在密码的板上 RAM 卸载（换出）到台式 PC 的硬盘上以便腾出空间来加载（换入）第一应用安全小程序到板上的 RAM。

然后，该 ROM 加载控制程序在加载第一应用安全小程序时予以检查。在确定该加载的第一应用安全小程序是有资格访问该密码单元资源以后，在该密码单元的微处理器运行该第一安全小程序，随后将该铁控制转移给该第一安全小程序。

当第一安全小程序完成时，该密码单元卸载（换出）加密形式的当前加载的第一安全小程序到 PC 的硬盘驱动器，并加载（换入）下一个安全小程序。每个安全小程序的密码内容被保存在储存在 PC 硬盘上的文件中。以那样的方式，单个密码单元在多个独立用户之中被共享。

如果遇到一个未知的应用安全小程序（即从未被加载到特定的密码单元的安全小程序），该 ROM 加载控制程序换回到自然上程序，它建立一个与 OPC 的安全通信会话。如果写此未知应用安全小程序的开发者以前被 OPC 授予加载和运行该安全应用程序的许可，则该密码单元将从 OPC 接收解密和运行该未知应用安全小

程序所必需的密码密钥。同时，OPC 在该小程序登录处记录该密码单元用户识别符，然后将该密码单元与该 OPC 授予加载及运行许可的安全小程序相关联。以后，该密码单元将加载与卸载该安全小程序而不必再与 OPC 通讯。

最后，当没有应用安全小程序与运行时，该 ROM 加载控制程序将自然形式安全小程序换回放入密码单元的板上 RAM。以这样的方式，每个单独的用户将密码单元使用于分别各自的安全应用程序。因此，使用多个单独的安全应用程序的多个单独的用户共享此密码单元。

附图概述

图 1 是按照本发明的一个公共密码系统的方块图。

图 2 是按照本发明的一个公共密码单元的广场图。

图 3A 是一个流程图，示出了按照本发明产生一个加密小程序的方法和装置。

图 3B 是一个流程图，示出了按照本发明一个加密的小程序并将该的小程序加载到一个公共密码控制系统的板上 RAM 中的方法和装置。

图 4 是用于在 PC 硬盘驱动存储器中储存一个加密的安全小程序的安全存储页格式的概图。

图 5 是一个流程图，示出按照本发明在密码运行中心一个安全小程序的台式开发者登录过程。

图 6 是一个流程图，示出按照本发明在密码运行中心一个安全小程序的台式 PC 机初始化过程。

图 7 是一个流程图，示出按照本发明由 O/S 的 ROM 加载控制程序部分做密码初始化的方法。

图 8A 是 O/S 的 ROM 加载控制程序部分的一个流程图，示出按照本发明从密码单元将一个安全小程序卸载（换出）到 PC 的方法。

图 8B 是 O/S 的 ROM 加载控制程序部分的一个流程图，示出按照本发明将一个安全小程序从 PC 加载（换入）到密码单元的方法。

图 9A 是一个方块图，示出按照本发明将对应于一个安全小程序的密码内容解密并从 PC 硬盘劝器加载（换入）到密码单元 RAM 存储器。

图 9B 是一个方块图，示了按照本发明将对应于一个安全小程序的密码内容加密并从密码单元 RAM 存储器卸载（换出到）到 PC 硬盘驱动器。

实施发明的较佳方式

系统操作

示于图 1 的一个公共密码系统的广场图包括一个密码运行中心 OPC21，一个台式 PC22，一个软件开发者 PC10 和一个发行媒体 20。软件开发者使用软件开发工具包建立安全小程序 14。设计安全小程序 14 实现给定的安全功能，作为主软件应用程序 16 的部分。软件开发者通过某个发行媒体 20 发行包括该加密的 18 安全小程序 14 的软件应用程序。

为了解密该安全小程序 14，在 PC10 的软件开发者通过如电话线调制解调器那样的达到 OPC21 的安全通讯链路发出一个请求 15。请求 15 包括实际提出的安全小程序 14。响应该请求 15，在 OPC21 审查安全小程序 14 是否符合安全标准。例如，提出的安全小程序 14 不应试图访问和输出被禁用的密码，不应搅乱内部的消逝时间计数器（一个安全时钟）或在密码单元设置许可位（下面讨论）以便授予自己访问敏感区。如果提出的安全小程序 14 由于某种原因不满足安全标准，它将被拒绝登录。

另一方面，如果 OPC21 认可安全小程序 14 登录，OPC21 将选择一个单独的序列号（S/N）17 及一个任意的编码密钥 C19 与该小程序相关联。S/N17 和编码密钥 C19 从 OPC21 通过请求 15 使用的同样的安全电话线调制解调连结通讯到软件开发者 PC10。OPC21 在一个小程序登录处 23 维持一个发出的 S/N 和对应的发出的编码密钥 C 的数据库。此提出的小程序就被正式登录，并由 OPC21 授予许可，能由任何公共密码控制单元使用（即运行）。

在 PC12 的软件开发者在一个小程序编码器 18 中使用接收到的编码密钥 C 进行处理，加密认可的安全小程序 14。在 PC12 的软件开发者还接收 S/N 序列号，在编码器 18 中处理以识别被认可的安全小程序 14。完成的安全小程序（使用编码密钥 C19 另密并使用 S/N17 识别）替代软件应用程序 16，并通过如软盘、CDROM、全球广播、卫星、有线电视等那样的发行媒体发行到台式 PC22。

在软件应用程序 16 安装到台式 PC22 以后，加密的安全小程序存入硬盘驱动器 26。硬盘驱动器 26 通常保存有多个加密的安全小程序 28，30，32，它们对应于在台式 PC22 上使用的多个软件应用程序。每个储存的小程序 32 包含一个识别的 S/N，如 S/N32A。台式 PC22 还包括标准 PC 的部件，如调制解调器 24，CPU34，ROM36，时钟 38，RAM40 通过标准 PC 总线 25 连续的输入／输出。此外，台式 PC22 包括一个具有单独单元识别（UID）44A 的密码单元，它也耦合到总线 25。

在运行时，首先存在台式 PC22 的硬盘驱动器 26 上的软件应用程序 16 需要执行加密的小程序 32，然后台式 PC22 建立与 OPC21 的一个安全通讯会话。台式 PC22 需要从 OPC21 来的许可，来使用加密的小程序 32。为获得许可，密码单元 22 将它的 UID44A 和安全小程序 32 的 S/N32A 发送到 OPC21。

OPC21 使用以前提供的单独的 S/N17 在小程序登录处 23 查找对应的随意提供

的编码密钥 C。而且，OPC12 通过另到小程序登录处 23 而进入事务（由密码单元 44 使用 S/N32A）。小程序登录处 23 记录了所有登录的安全小程序 S/N，对应于每个 S/N 的编码密钥 C，和所有被授予许可运行每个对应的安全小程序的密码单元 UID。例如，登录处 23 示出，对应于密码密钥 C＝Z 的加密小程序 S/N32A 已被登录，且那于 UID＝44A 的密码单元已被授予许可来解密并执行（运行）带 S/N＝32A 的被登录的小程序。

公共密码控制单元

在图 2 中的公共密码控制单元包括微处理器 206，RAM 存储器 222，224 和 ROM 存储器 208。分配 RAM 存储器作为签名寄存器 224 和主密码程序控制 222 区的存储。RAM208 包含 O/S 的加载控制程序部分。包括在密码单元 44 的仍然是一个 DES 引擎 218，一个非易失性存储器 220，一个消逝时间（实时时间）计数器 204 和访问控制寄存器 212。对了在密码单元 44 和主 PC 机之间的通讯提供一个主（台式）PC 机接口 202。在密码单元内部的通讯是在密码单元 44 中部件之间通过地址和数据的通用数据总成 210 上提供的。

DES 引擎 218 方便了在密码单元 44 的保护环境中的密码操作。例如，内部的非易失性存储器 220 提供了密码密钥的安全存储。的时间计数器 204 使得在密码单元 44 的安全环境中保护时间和日期的计算不被扰乱。一些重要的操作，如读写消逝时间计数器 204，访问和改变在非易失存储器 220 中的密码存储的内容，访问和改变签名寄存器 224 的内容被硬件所限止。尤其是，访问密码单元由设置访问寄存器 212 的单独许可位 216（下面讨论）所控制。

访问寄存器 212 包括专门的保护特征以保护加载到程序控制 RAM222 的安全小程序损害密码单元 44 的安全特征。尤其是，访问寄存器 212 包括一个许可寄存器，而且访问寄存器 212 的单独的许可位 216 确定了给定的安全小程序允许访问哪些资源。例如，硬件连接信号（许可控制）226 提供了硬线连结的限止，加载在 RAM222 中的给定的安全小程序是否允许访问所有签名寄存器 224，消逝时间计数器 204 和客户密钥及安全密码存储区 220 或其中一部分。秘密的客户密钥对每个密码单元是单独的，且在制造时与其他密码密钥一起存入非易失性存储器 220。

仅当从访问寄存器 212 授予许可时，在 RAM222 中给定的安全小程序才允许访问（读或写）重要的操作。许可寄存器从解密安全小程序由加载控制 ROM208 程序加载。作为防止非授权访问的进一步措施，许可寄存器 216 可以只由加载控制 ROM208 的指定执行来访问。一旦许可寄存器被写入新值，就执行地址控制 214。尤其是，只有当地址检测指示出加载控制 ROM 正执行许可位加载，从地址检测 214 的写使能信号才被激活。以这样的方式，许可寄存器 216 只能由加载控制 ROM208

的适当的指令序列加载。因此，在 RAM222 以外运行的安全小程序不能改变许可寄存器 216 的许可位。

访问寄存器的许可位

访问寄存器 212 的单独许可位 216 提供对消逝时间计数器 204 的控制。尤其是，一个许可位控制消逝时间计数器 204 是否可读，另一个许可位控制消逝时间计数器 204 是否可写。只有 OPC 通过自然形安全小程序得到许可（通过许可位的设置）将值写入消逝时间计数器。

访问寄存器 212 的单独许可位 216 提供对在非易失存储器 220 中的客户密钥和安全密钥存储的控制。尤其是，一个许可位确定该小程序是具有读（不是写）客户密钥的访问。在客户密钥是工厂安装的，不能改变。此客户密钥可由软件开发者用于与安全小程序相关的密码计算。

储存在非易失性存储器 220 的其他密钥包括对特定软件开发者的私人密钥。即，特定的软件开发者可以不用共享的客户密钥。相反，对那样的特定一个私人密钥。在那样的情况，对应于一个专门的私人密钥的许可位允许由特定的软件开发者的安全小程序访问专用的私人密码。由其他软件开发者的安全小程序将不对那个专用的私人密钥设置许可位，因而不具有访问此坟门的私人密钥的资格。此外，访问寄存器 212 的分别的许可位 216 确定该 安全小程序是否可以写入一个新的专门的私人密钥来覆盖在非易失性存储器 220 中老的专人仫人密钥。除了专门的私人密钥以外，非易失性存储器 220 可以存储用于鉴别一个公共私人密钥对的公共密码部分的数字检验证。

访问寄存器 212 的单独许可位 216 与 RAM 的签名登记部分 224 结合被用于提供有关哪个安全小程序可以加载到密码或从中卸载的访问控制尤其是，在签名登记的一个入口设置一个删除标记将删除此的安全小程序。其后密码片将不加载或卸载在签名登记中由删除标记指明的安全小程序。最后，OPC 可以通过设置抑制整个密码单元。被抑制的密码单元 44 的适当的许可位 216 来抑制整个密码单元.被抑制的密码单元 44 不能运行任何安全小程序,除非该密码单元 44 被 OPC 重新激活。

安全小程序登录

如上指出的，出应用程序开发者将安全小程序作为主应用程序的一部分来设计。写此安全小程序专门在密码单元 44 上运行。安全小程序必须压缩到足以能存入密码单元的板上 RAM（图 2 中的 222）。大得不能存入板上 RAM 的安全应用程序可以分成两部分，即分成两个安全小程序。在安全小程序能与主应用程序一

起发行并在密码单元上运行以前，开发者必须用 OPC 登录此安全小程序。如上指出，开发者建立了与 OPC 的安全通讯会话。一个适合与 OPC 安全通讯的系统示于美国专利 5，615，264，5，761，283 和 5，764，762。

图 5 示出在 OPC 上开发者的登录过程。在步骤 5100PC 接收一个安全小程序登录的请求。此请求包括实际指出的安全小程序。在步骤 5120PC 审查提出的安全小程序是否符合适当的密码标准。例如，提出的安全小程序可以不试图揭示对每个独立的密码单元是单独的客户密钥或任何其他安全密钥。既不能输出密码，也不能输入附加的密码，象在变址程序循环中那样，间接程序跳转是一个安全的危险。通过系统的安全性受分割的经验，可以设计许多测试以保证所提出的安全小程序是安全且恰当地设计的。如果该提出的安全小程序不能通过任何一个测试，在步骤 5120，PC 拒绝登录该提出的安全小程序。

如果所有的测试通过，在步骤 5140PC 选择一个序列号 S/N 和一个编码密钥 C。在步骤 514，OPC 还将 S/N 和编码密钥 C 输入到一个小程序登录（图 1 中的 23）中。在步骤 516，通过将对新登录小程序的 S/N 和编码密钥 C 发送到软件开发者而完成了登录过程。

使用的密码惯例

图 3A，3B，9A 和 9B 示出用符号表示的密码操作。如在此使用的，对加密和解密的较佳过程是数据加密标准（DES）。

简而言之，对 DES 的电子码书方式（Ele-ctronic code book mode - ECB），一个 64 位（8 个字节）的输入块按照 56 位码被转换成一个 64 位的输出块。对解密，完成逆过程，将 64 位输入住使用同样的 54 位码转换成 64 位输出位。通常 DES 密码以 64 位，8 个字节表示，每字节具有 7 位另 1 个奇偶校验位，或 56 密码位加 8 个奇偶校验位。

如在这里使用的，借助一个秘密密钥对一个变量完成一个密码操作意味着使用一个秘密密钥产生另一个密钥来加密（或解密）该变量。加密可以在一个单个密钥或多个密钥，如三个密钥的组的基础上完成。除非特别指出，加密或解密意味着在三个密钥的组的基础上 DES 加密或解密的 ECB 方式，对于三个密钥的加密，使用三个密钥（密钥 1，密钥 2，密钥 3）的组如下采用 DES 另密一个变量：用密钥 1 加密，用密钥 2 解密，并用密钥 3 加密。三密钥的解密是个逆过程一用密钥 3 加密，用密钥 2 加密，再用密钥 1 解密。CBC 意义是使用初始向量 IV，DES 标准的密码块连续模式（Cipher block chaining mode）。除非另加说明，对一个 CBC DES 加密或解密的 IV 将是 0。

密码单元初始化及登录

图 7 示出通过 ROM 加载控制程序密码单元初始化和登录的方法。在上电时，ROM 另载控制程序（在图 2 的 ROM208 中）在步骤 710 将一个初始自然形式安全小程序从硬盘驱动器（图 1 中的 26）加载到板上的 RAM（图 2 中的 222）中。ROM 加载控制程序认为初始自然方式安全小程序是预先认可并用固定密钥加密的。初始自然形式安全小程序通过将访问寄存器 216 的所有许可位均置成使能被授予对整个密码单元的资源访问的权限。加载以后，密码单元的控制被转移到刚加载到板上 RAM 的初始自小程序。

如果这是首次使用密码单元，在步骤 712 开始一个登录过程。在步骤 714 建立个 OPC 的安全通讯会话，且密码单元与 OPC716 进入登录过程。登录包括输入识别用户的数据（名字、地址等）并将与密码的 UID 相关的用户数据转送到 OPC。在步骤 714 与 OPC 通讯会话的过程中，OPC716 有机会下载任何程序的改变来更新初始自然形式安全小程序。在登录过程结束后，程序控制返回到台式 PC。然后，密码单元进入等待状态，直到台式 PC 准备好加载拟在密码单元运行的第一个安全小程序。

登录的安全小程序的加密

安全小程序要加密，图 3A 是用于安全小程序加密的加密密钥组的流程图。软件开发者从所希望的安全小程序 322 开始。如上指出，安全小程序 322 先前已被软件开发者发送到 OPC，且小程序的 S/N320 和编码密钥 C318 先前已作为小程序登录过程的部分被接收。

软件开发者选择了一个在步骤 302 自己挑选的编码密钥 A（编程者密钥）。然后，密码密钥 A 在加密器 304 中借助编码密钥 C 加密，形成加密的编码密钥 A’。安全小程序 322 是在加密器 324 中在编码密钥 A 下用了密码 CBC 加密。在加密器 326 中计算一个消息鉴别码（message authentication code - MAC）。MAC（也称为操纵检测码）是在加密的包后面的数字签名，它由该加密包的接收者检查，以难加密包的内容未被修改。在步骤 306，MAC 通过将 S/N320，编码密钥 A’ 和从加密器 324 的输入来的加密安全小程序组合到一个安全包中而产生的。

组合一个安全包 306 的目的是在加密器 326 中产生 MAC316，并将其在安全包的后面形成一个安全页。开发者的 MAC 密码是在加密器 328 中借助编码密钥 C318 由加密 S/N320 而形成。MAC 签名本身通过在安全包 306 上用三密钥 CBC 加密 326 而产生。尤其是加密器 326 的输出的最后部分形成 MAC 签名 316，它在安全包 306 之后。

计算的 MAC 与安全包结合形成一个安全页 306，它从密码单元输出，并最终

存入主 PC 机的硬盘驱动器。

为将加密的安全小程序存入 PC 的硬盘驱存储动器，安全存储页的格式示于图 4 中。安全存储页从安全包（S/N310 后面是编码密钥 A’ 312，再后面是加密的安全小程序 314）开始，由计算 MAC 结尾。

安全小程序的初始加载的解密

密码单元一个开始遇到的加密安全小程序，如在图 3B 的加密密钥组流程图所示。因为这是一个以前尚未运行的安全小程序的初始加载，在 RAM224 的小程序签名登录部分找不到 S/N310（在签名登录 224 中找到 S/N 的情况该加密的小程序以前已被运行，可应用图 9A）。如上面指出，对一个开始遇到的安全小程序，自然形式的安全小程序将 S/N338 发送到 OPC，并从 OPC 接收编码密钥 C336。

首先，在解密器 330 中借助密码密钥 C336 通过解密编码密钥 A’ 312 恢复软件开发者的编码密钥。加密的安全小程序 314 是在解码器 332 中借助从解码器 330 的输出得到的恢复的编码密钥 A 被三密钥 CBC 解码的。对安全包（S/N310，编码密钥 A’ 312 和加密的安全小程序 314）的 MAC 是借助开发者的 MAC 密钥在三密钥 CBC 加密器 340 中被计算。开发者的 MAC 密钥借助于在加密器 348 的编码密钥 C336 通过加密 S/N338 计算的，336 被耦合到加密器 340 的密码输入。

在加密器 340 的输出处的计算的 MAC 在比较器 342 中与接收到的 MAC316 比较。如果计算的 MAC 和接收的 MAC 相当等 334，则与（AND）门 334 使能，且在解码器 332 的输出处的解密的安全小程序被存入板上 RAM 的密码控制部分 222。但是，如果计算的 MAC 和接收的 MAC 不等 346，则该安全小程序不允许加载到板上 RAM222 上，且不允许运行。相反，与门 332 不使能，而且在解密器 332 输出处的解密的安全小程序不被存入板上 RAM 的密码控制部分 222。一个错误消息返回到台式 PC。

在安全小程序加载时的 OPC 控制

本系统给出不管安全小程序能否加载到给定的密码单元时的 OPC 控制。图 6 示出在 OPC 的初始加载控制过程。在步骤 610 从 OPC 处的台式 PC 接收 S/N 以后，OPC 在步骤 612 检查，该小否具有合法的 S/N。如果没有，OPC 返回一个错误消息，请安全小程序是“不合法（INVALID）的”。在步骤 614，OPC 检查，如果 S/N 碑是合法的，是否已被删除。如果确实如此，OPC 返回一个错误消息，该安全小程序已被“删除（CANCELLED）”。在步骤 6160PC 检查，由其 UID 识别的给定的密码单元是否允许加载此特定的安全小程序。 是，OPC 返回一个错误消息，该安全小程序的加载是“不允许（DISALLOWED）”。如果 S/N 是合法的，未被删除，且密码单元允许加载此安全小程序，则在步骤 618 在 OPC 处的小程序登录中查找编

码密钥 C，并发送到密码单元。

以这样的方式，在初始安全小程序安装过程中 OPC 维持控制。例如，如果一个安全小程序已被改写以修正一个问题，OPC 将不允许以后的用户将早期的版本安装到密码单元。如果一个给定的密码单元已知被泄密了，对此密码单元 UID 不再允许安全小程序的加载。

ROM 加载控制 O/S一密码内容的交换

密码单元的 ROM 加载控制程序（O/S）支持多个同时的用户。为了在用户之间切换，当前安全小程序的密码内容从密码单元卸载，并存入台式 PC 的硬盘驱动器。然后通过从台式 PC 的硬盘驱动器提取以前运行的安全小程序的以前存入的密码内容，此密码被恢复到对应于那个以前运行的安全小程序的以前的密码状态。如果这里使用，术语“加密的安全小程序”，“密码内容”和“在其密码内容中（带有或包括）的加密的安全小程序”都认为是基本上等价的术语。

在本实施例中，软件开发者配置该安全小程序在该程序存在以前将密码参数存入板上 RAM 的密码程序控制部分 222。软件开发者期望那个安全为其安全应用程序所需，并需要将密码单元恢复到它的以前密码状态，并继续此安全应用程序。

在某些安全应用程序中，需要密码单元的所有密码参数来恢复该密码单元。在另外一些安全应用程序中，只需密码的一个子集。在又一个实施便中，密码单元自动地将它本身（密码单元）的整个密码状态存入与每个安全小程序有关的一个单独的文件。在后面的情况，切换密码状态（存储和恢复密码状态）的重担自动地由密码单元的操作完成而不必要开发者的软件干预。

在本实施例中，对每个给定的安全小程序的密码内容文件程序加上密码单元的密码状态。下面给出密码内容的格式：

表 I一密码内容（29K）

明文表头 序列号（S/N），大小 修订版本号，时间标记
程序数据一安全小程序
永久性寄存器存储
堆（暂时存储）
堆栈
MAC / 签名

除了明文表头外，密码内容是加密的。明文表头由下列字段组成：

序号号（S/N）：S/N 是在登录过程中为该安全小程序颁发给软件开发者的初

始序列号。

大小：对应于从密码单元卸载并存入硬盘驱动器的密码内容的字节数。

修订版本号：用于跟踪对初始登录的安全小程序的变化。

时间标记：对应于在卸载时间密码单元实时时钟的内容。

密码内容的另密部分由下列字段组成：

程序数据：包括任何在程序执行过程中作的修改的安全小程序。

堆（暂时存储）：表示在卸载以前密码的密码状态的参数。

堆栈：程序的堆栈存储，如对嵌套子程序的返回地址。

MAC／签名：对整个密码内容计算的 MAC。

表 II—签名登录

板上 RAM 的签名登录 224 具有下述格式：

序列号（S/N）	MAC（签名）	标志
S/N 1	MAC 1	标志 1
S/N 2	MAC 2	标志 2
---	---	---
S/N 31	MAC 31	标志 3

S/N：小程序的序列号。

MAC：对储存在 PC 硬盘驱动器的小程序的密码内容的消息鉴别码。

标志：储存在签名登录中的标志包括一个小程序删除标志，它由 OPC 设置以防止被删除的小程序又一次的使用。

图 8A 的 ROM 加载控制 O/S—换出

在图 8A 中示出 ROM 加载控制程序（O/S）的换出部分的流程图。操作系统的换出部分的功能是卸载当前在密码单元运行的安全小程序到台式 PC 的硬盘驱动器，小程序包括其密码内容。例如，安全小程序可以具有内部寄存器存储，堆栈指针和其它程序，它们在执行时被修改，并组成它的密码内容的部分。

在图 8A 中，在步骤 810 若当前的安全小程序已经完成，在步骤 812 将密码单元的密码状态保存在板上 RAM 中。储存的密码状态包括 DES 引擎（图 2 中的 218）的状态和将密码恢复到它当前状态所需要的任何其他变量。然后，在步骤 814RAM 的内容（按照在图 9B 中示出的加密密码组）被加密。在步骤 816 对包括明文表头和 S/N 的加密 RAM 内容计算 MAC，且在步骤 818 将此 MAC 存入（或更新）RAM 签名登录 224。在步骤 820 组成安全页并在步骤 882 存在如式 PC 的硬盘驱动器。

图 8BROM 加载控制 O/S 0 换入

在图 8B 中示出 ROM 加载控制部分（O/S）的换入部分的流程图。操作系统的这部分的功能是将下一个安全小程序加载到板上 RAM 以便在密码单元运行，包括从台式 PC 的硬盘器恢复可能有的相应的以前的密码内容。

在图 8B 中，到了加载一个安全小程序到板上 RAM 的时信，ROM 另载控制程序首先在步骤 830 检查，S/N 是否在 RAM 的签名登录部分（图 2 中的 224）。S/N 是否在签名登录 224 决定了此密码单元以前是否运行过此特定的安全小程序。

如果该小程序的 S/N 未在登录处，密码单元未曾运行过此特定的安全小程序。然后程序在步骤 834 检查，以确定签名登录是否已满或者还有空间用于另外的存入内容。如果签名登录已满，返回一个错误消息“登录已满—REGISTRY FULL”。如果签名登录未满，ROM 加载控制程序在步骤 838 换入自然上程序，如上按照图 6 所述，它建立了一个与 OPC 的通讯。

如上结合图 6 指出，在步骤 838 自然形式安全小程序将提出的安全小程序的 S/N 送到 OPC，并在步骤 839 获得编码密钥 C。在上面还提出，密码单元使用接收的编码密钥 C 在步骤 837 提出的安全小程序并计算对该安全小程序的 MAC。对加载到给定的密码单元的安全小程序的解密密钥首先结合图 3B 在上面叙述。

如果密码单元以前已经运行过此特定的安全小程序，则在步骤 830 将在登录中找到此 S/N。在此情况，在步骤 832 从 RAM 的签名登录部分（图 2 中 224）取出 MAC。在步骤 836 解密此安全小程序（用其密码内容）并计算 MAC。用于解密存储的小程序并计算 MAC 的密钥组在下面结合图 9 描述。

在换入过程的这个阶段，有 3 个与该安全小程序相关的 MAC（带有它的密码内容），ROM 加载控制程序（O/S）试图将其加载到密码单元的板上 RAM。第一个 MAC 从签名登录取出，第二个 MAC 与储存的密码内容一起从台式 PC 接收到，第三个 MAC 根据输入的解密的小程序计算。如果在步骤 840 判断 3 个 MAC 互相相等，则解密的安全小程序被加载到 RAM 的密码程序控制部分，且在步骤 842 开始执行该安全小程序。否则，从步骤 840 返回给 PC 一个错误消息“访问被拒绝—ACCESS DENIDE”。

密码内容文件

图 9A（换入）和图 9B（换出）示出为了安全小程序（在对应的密码内容中）在 RAM222 的密码程序控制部分和台式 PC 的硬盘之间交换的相应的解密和加密密钥组。尤其是，图 9A 是一方块图，示出用于将对应于一个安全小程序的密码内容解密并从 PC 硬盘驱动器加载（换入）到密码单元 RAM 存储器的方法和装置。图 9B 是一个广场图，示出用于将对应于一个安全小程序的密码内容加密并从密码单元 RAM 存储器卸载（换出）到 PC 的硬盘驱动器的方法及装置。

密码内容换出一图 9B

在图 9B 中，RAM222 的密码程序控制部分的内容作为加密的文件 962A 被卸载到硬盘驱动器 26 中。RAM 存储器的签名登录部分未被卸载。所产生的各种加密密钥是基于第一个固定的字符串 A940，第二个固定的字符串 B956 和一个称为客户密钥 942 的秘密密钥。 客户密钥 942 被存入可编程存储器（图 2 中的 220）。客户密钥存储器通常是非易失的，可以由任何适合的非易失存储器实现，如可熔断的连结。EEPROM，电池供电的 RAM 等。对每个密码单元所存储的客户密钥 942 是唯一的，且在制造时安装。

第一个固定字符串 A940 是在加密器 944 中借助客户密钥加密的。加密器 944 的输出用作加密器 946 的密钥，加密将被卸载的小程序的 S/N（明文）。加密器 946 的输出用作为密钥在 3 密钥 CBC 加密器 948 中加密此安全小程序。注意，对换出的安全小程序的加密密钥（到加密器 948）与该安全小程序初始加载使用的不是同一密钥。对于安全小程序的初始加载，使用的密钥是开发者的编码密钥 A。在图 9B 中用于卸载的密钥是固定字符串 A940，S/N 和客户密钥 942 的函数。因为每个客户官衔每个密码单元是单独的，存在硬盘驱动器 26 的换出密码内容不能换回到另一个密码单元。即，一旦一个安全包使用一个客户密钥被换出密码单元到硬盘驱动器，换出的安全小程序（在其密码内容中）不能加载到具有不同客户密钥的不同的密码单元。

为了对密码内容（包括安全小程序）产生一个 MAC，组合一个安全包 950。安全包 950 包括明文的 S/N 和加密的安全小程序（带着它的密码内容）。MAC 通过借助由加密器 954 的输出导出的密钥的 3 密钥 CBC。加密安全包而产生。如从图 9B 所见，从加密器 954 输出的 MAC 密钥是固定字符串 B956，（通过加密器 944 和 946），S/N，客户密钥 942 和固定字符串 A940 的函数。

尤其是，加密器 946 的输出作为加密密钥是到加密器 954 的输入，后者加密固定字符串 B 成为到了密钥 CBC 回避顺 952 的 MAC 密钥。在加密器 952 的输出端的 MAC 与安全包组合形成一个安全页 958。安全页 958 是与另外的密码内容 962N 以及换出的自然形式小程序 960 一起存在硬盘中的 962A。

密码内容换入一图 9A

当密码单元在多个同时的安全应用程序之间切换时，在图 9A 中以前存入的密码内容 912，918A - 918N 从硬盘驱动器 26 加载到板上 RAM 的密码程序控制部分 222。密码单元使用签名登录的内容来确定每个以前储存的密码内容 912，918A - 918N 是否允许加载和运行。在密码内容 912 中的自然形式安全小程序与其他由密

码单元运行的多个同时的安全的小程序 918A － 918N 以同样的方式换入或换出密码单元。

图 9A 中的密钥组（换入）完成在图 9B（换出）中密钥组的逆向密码过程。尤其是，固定字符串 A910 在加密器 916 中借助于客户密钥被加密。加密器 916 的输出被用作加密器 920 中的密码来加密将被加载的安全小程序的 S/N 和密码内容 918A。加密器 920 的输出被用作小程序的解密密码，未在 3 密钥 CBC 解密器 922 中解密安全小程序密码内容。对安全小程序换入的小程序解密密钥（到解密器 922）是在换出过程中用来加密安全小程序的同一个密钥。

对密码内容 918A 的 MAC 密钥是通过在加密器 932 中借助于小程序解密密钥（加密器 920 的输出）的第一个加密固定字符串 B930 计算的。然后加密器 932 的输出被用作在加密器 926 的密钥，以形成有关密码内容 918A 的安全页部分的计算的 MAC，为检查此 MAC，从 RAM 的签名登录部分 224 取出储存的 MAC。然后在比较器 928 中比较从加密器 926 输出计算的 MAC，从签名登录 224 储存的 MAC 以及从密码内容 918A 接收的 MAC 等 3 个 MAC，在步骤 934 如果 3 个 MAC 相等，则与（AND）门 924 使能，将接收到的安全小程序加载到 RAM 的密码程序控制部分 222。若在步骤 936，3 个 MAC 中任何一个与另一个不等，则与门 924 不使能，不能将接收到的安全小程序加载到 RAM 的密码程序控制部分 222。

安全小程序交换

安全小程序可以通过一步过程或二步过程换入到密码单元。二步过程已在上面描述。即，在将加密的安全小程序加载到板上 RAM 的密码程序控制部分以前，ROM 加载控制程序对其审查。在二步实现过程中，如果所有 MAC 答名测试通过，该安全小程序然后在第二步被解密并加载到板上 RAM。如果在第一步不允许加载，在第二步没有任何安全小程序的部分被加载到板上的 RAM。

在一步实施过程中，ROM 加载控制程序审查一个加密的安全小程序，同时解密并将解密的安全小程序加载到板上 RAM 的密码程序控制部分。如果 MAC 签名测试失败（在图 8B 中步骤 840），密码单元的控制不转移到刚加载的安全小程序。而是，下一个安全小程序或自然形式安全小程序被加载到 RAM 的该密码程序控制部分，履盖以前加载的未允许的安全小程序。但是，如果 MAC 签名被通过，则 ROM 加载 密码单元的以刚加载的安全小程序。

二步实施例通常更安全，因为在所有 MAC 签名的测试完成以前没有任何部分的新安全小程序被加载到 RAM222 的密码程序控制部分，一步实施时通常导致更快的安全小程序交换，因为新的安全小程序在板上 RAM 开始执行而不必等待第二步。

由 OPC 监控密码单元

在图 1 中的密码单元 44 由 OPC21 周期地览监控。即，至少每月一次或隔选定的时间间隔一次密码单元 44 发起一次与 OPC21 的通信佳话。通信可以通过调制解调器 24 到拔号连结。或通过 TCP/IP 协议通过因特网连接。在两者情况，密码单元 44 的状态被报告到 OPC21。周期通信的目的是将密码单元 44 的内容与在 OPC21 所期望的内容相同步。

例如，在与 OPC21 周期通讯期间，消逝时间计数器 204（图 2）与它期望的值校验。若需要使其同步。消逝时间的较大差异可能是是搅乱的指示，并可能导致由 OPC 抑制密码单元。在访问寄存器 212OPC 可以设置一个或多个许可位 216，来抑制密码单元。一般被抑制，抑制制的密码单元可以不加载或运行任何的安全小程序。

而且在与 OPC21 周期性通讯期间，签名登录（图 2 中 224）被检查以回顾哪个安全小程序已被加载到该密码单元并在其中运行。如果一个安全小程序已被删除（即，运行那个安全小程序的系统的广泛的许可已被撤消），将在答名登录中设置对应于那个安全小程序的删除标志，与许可控制（图 2 中 226）相结合，密码单元 44 将不换入（加载）该被删除的安全小程序。

说　明　书　附　图

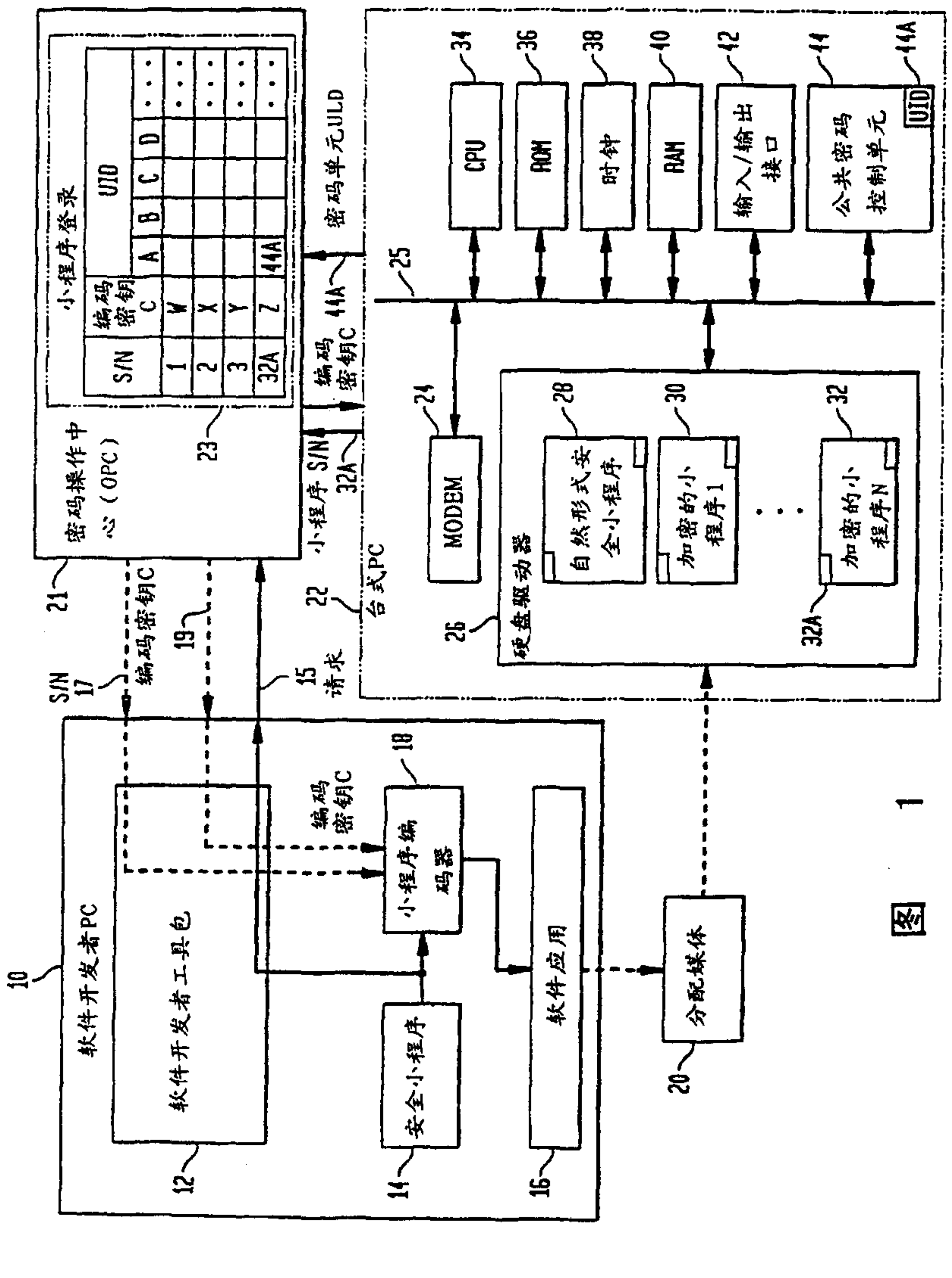

10
软件开发者PC
软件开发者工具包
12
安全小程序
14
小程序编码器
18
软件应用
16
分配媒体
20
编码密钥C
S/N
17
编码密钥C
19
15
请求
21
密码操作中心（OPC）
23
小程序登录
S/N
编码密钥C
UID
A
B
C
D
1
W
2
X
3
Y
32A
Z
44A
小程序 S/N
32A
编码密钥C
44A
密码单元ULD
22
台式PC
25
34
CPU
36
ROM
38
时钟
40
RAM
42
输入/输出接口
44
公共密码控制单元
44A
UID
24
MODEM
26
硬盘驱动器
28
自然形式安全小程序
30
加密的小程序1
32
加密的小程序N
32A

图　1

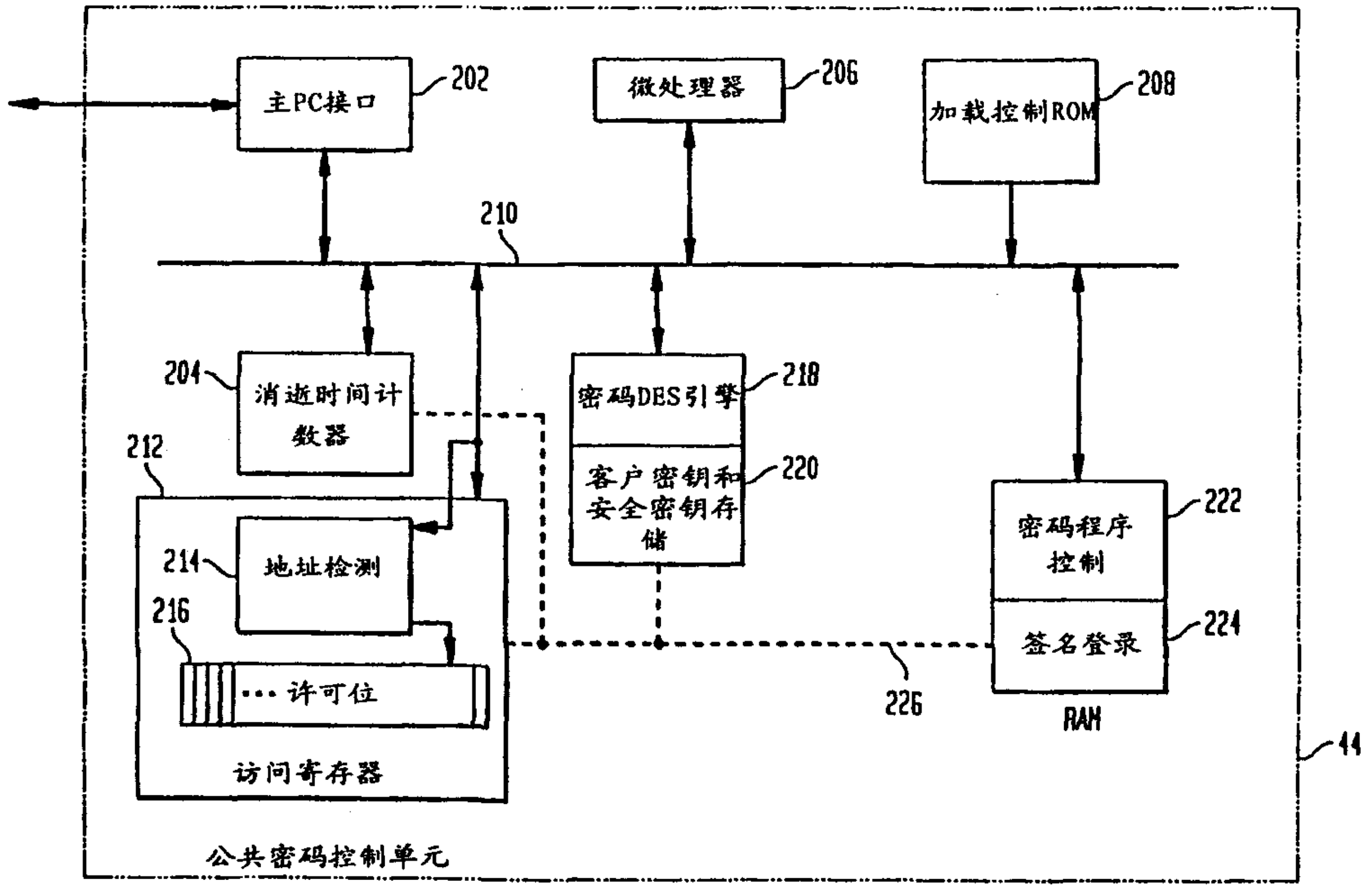
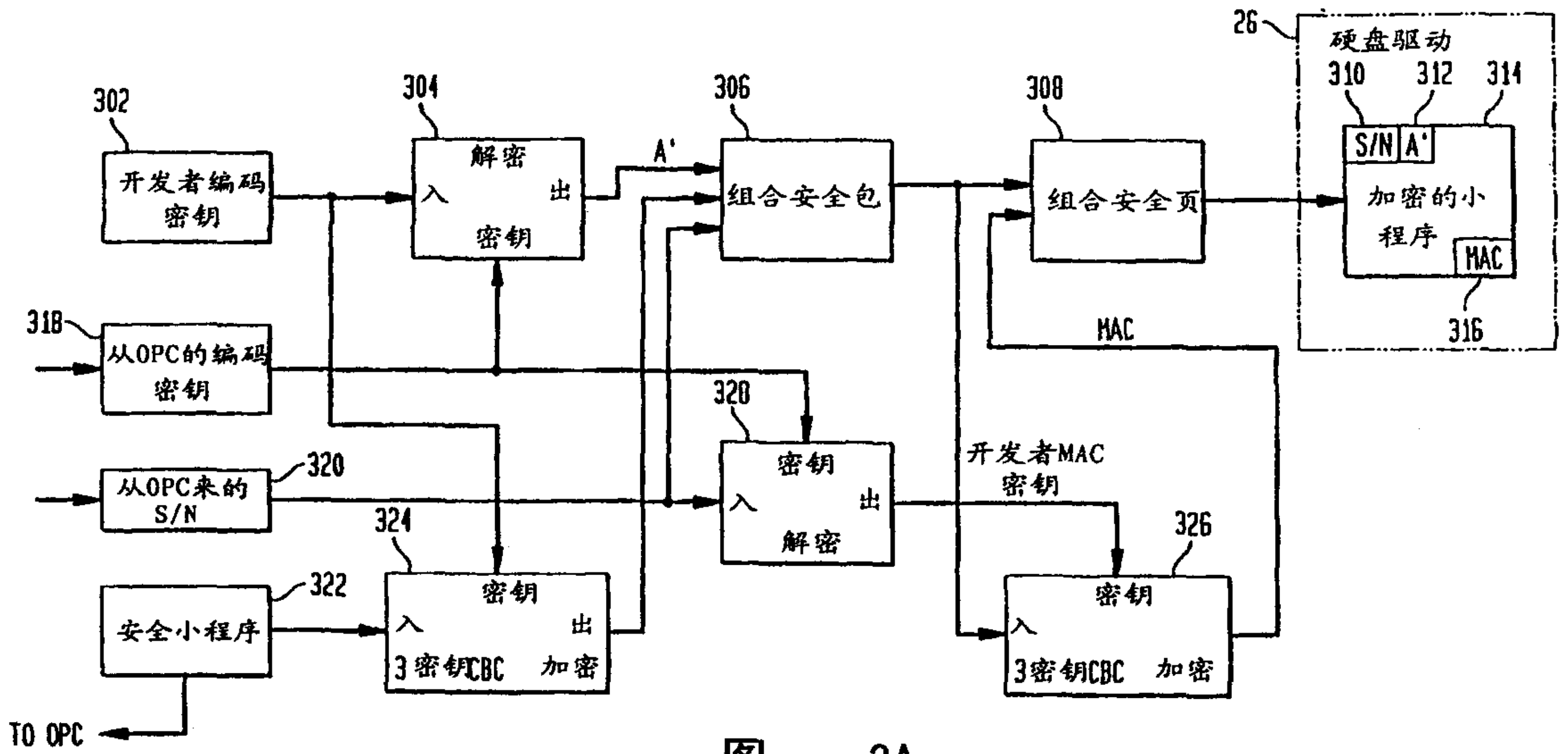
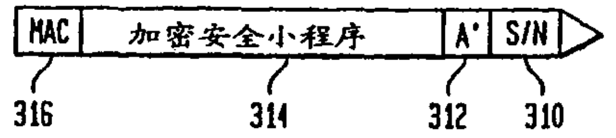

主PC接口
202
微处理器
206
加载控制ROM
208
210
204
消逝时间计数器
密码DES引擎
218
客户密钥和安全密钥存储
220
密码程序控制
222
签名登录
224
RAM
212
214
地址检测
216
… 许可位
访问寄存器
226
44
公共密码控制单元

图 2



26
硬盘驱动
310
312
314
S/N
A'
加密的小程序
MAC
316
302
开发者编码密钥
304
解密
入
出
密钥
A'
306
组合安全包
308
组合安全页
318
从OPC的编码密钥
320
从OPC来的S/N
320
密钥
入
出
解密
开发者MAC密钥
MAC
322
安全小程序
TO OPC
324
密钥
入
出
3密钥CBC
加密
326
密钥
入
3密钥CBC
加密

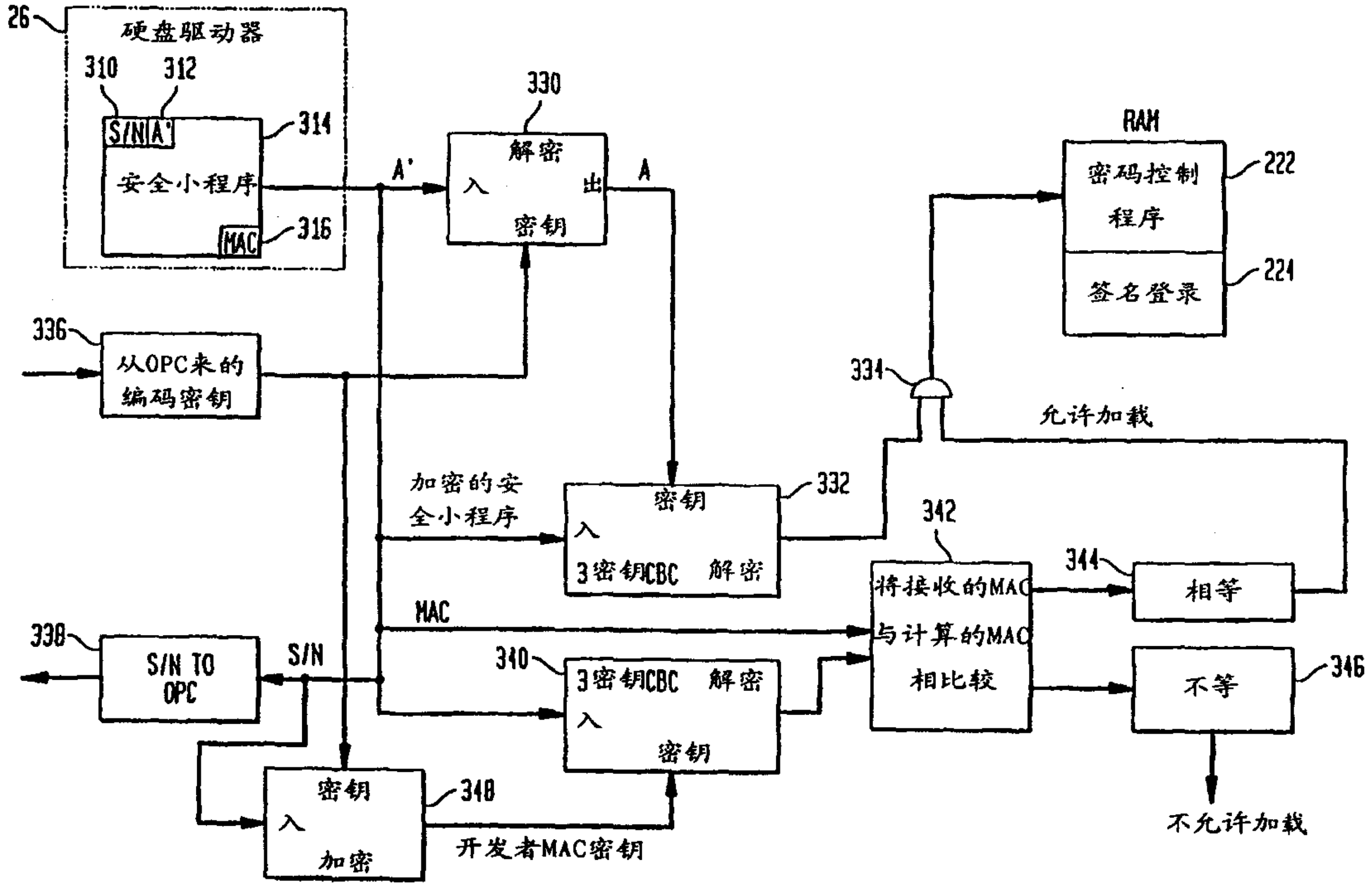
图 3A



MAC
加密安全小程序
A'
S/N
316
314
312
310

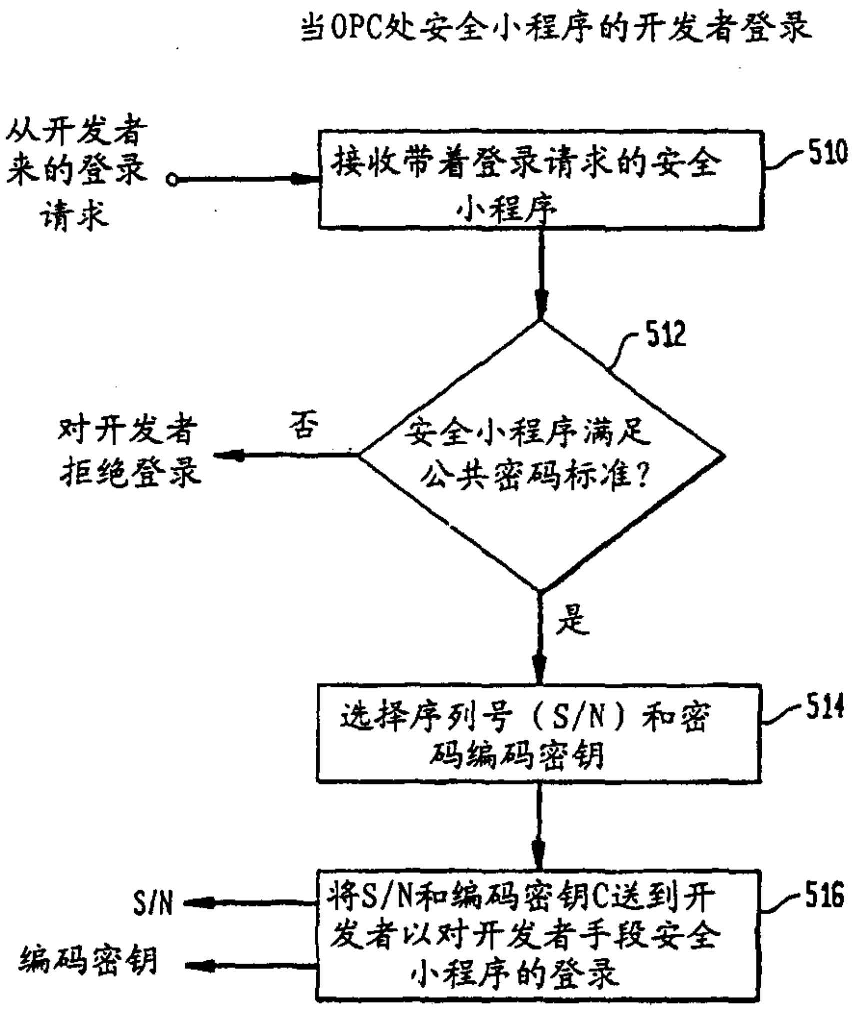
图 4

01.11.22



26
硬盘驱动器
310
312
S/N
A'
安全小程序
314
MAC
316
A'
330
解密
入
出
密钥
A
336
从OPC来的编码密钥
加密的安全小程序
332
密钥
入
3密钥CBC
解密
MAC
338
S/N TO OPC
S/N
340
3密钥CBC
解密
入
密钥
348
密钥
入
加密
开发者MAC密钥
334
允许加载
RAM
密码控制程序
222
签名登录
224
342
将接收的MAC与计算的MAC相比较
344
相等
346
不等
不允许加载

图 3B



当OPC处安全小程序的开发者登录
从开发者来的登录请求
接收带着登录请求的安全小程序
510
512
安全小程序满足公共密码标准?
否
对开发者拒绝登录
是
选择序列号（S/N）和密码编码密钥
514
将S/N和编码密钥C送到开发者以对开发者手段安全小程序的登录
516
S/N
编码密钥

图 5

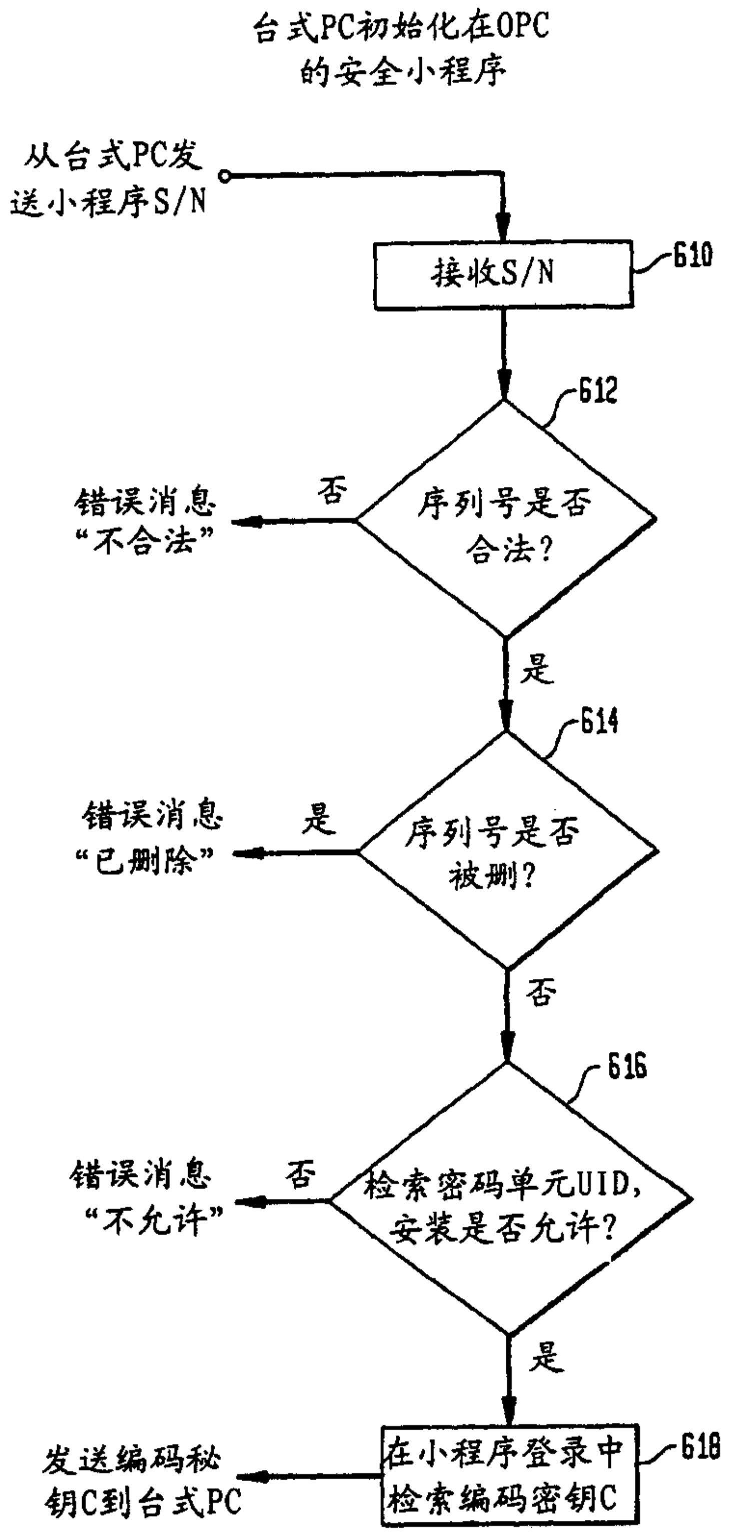

台式PC初始化在OPC
的安全小程序
从台式PC发
送小程序S/N
接收S/N
610
612
序列号是否
合法?
否
错误消息
"不合法"
是
614
序列号是否
被删?
是
错误消息
"已删除"
否
616
检索密码单元UID,
安装是否允许?
否
错误消息
"不允许"
是
在小程序登录中
检索编码密钥C
618
发送编码秘
钥C到台式PC

图 6

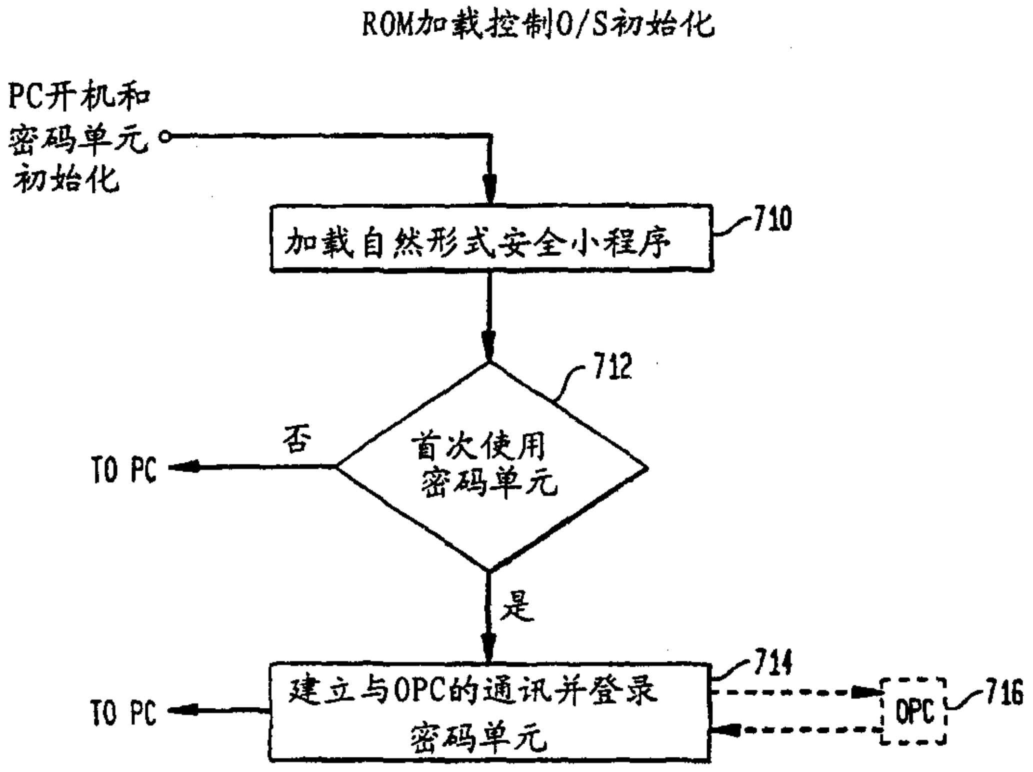
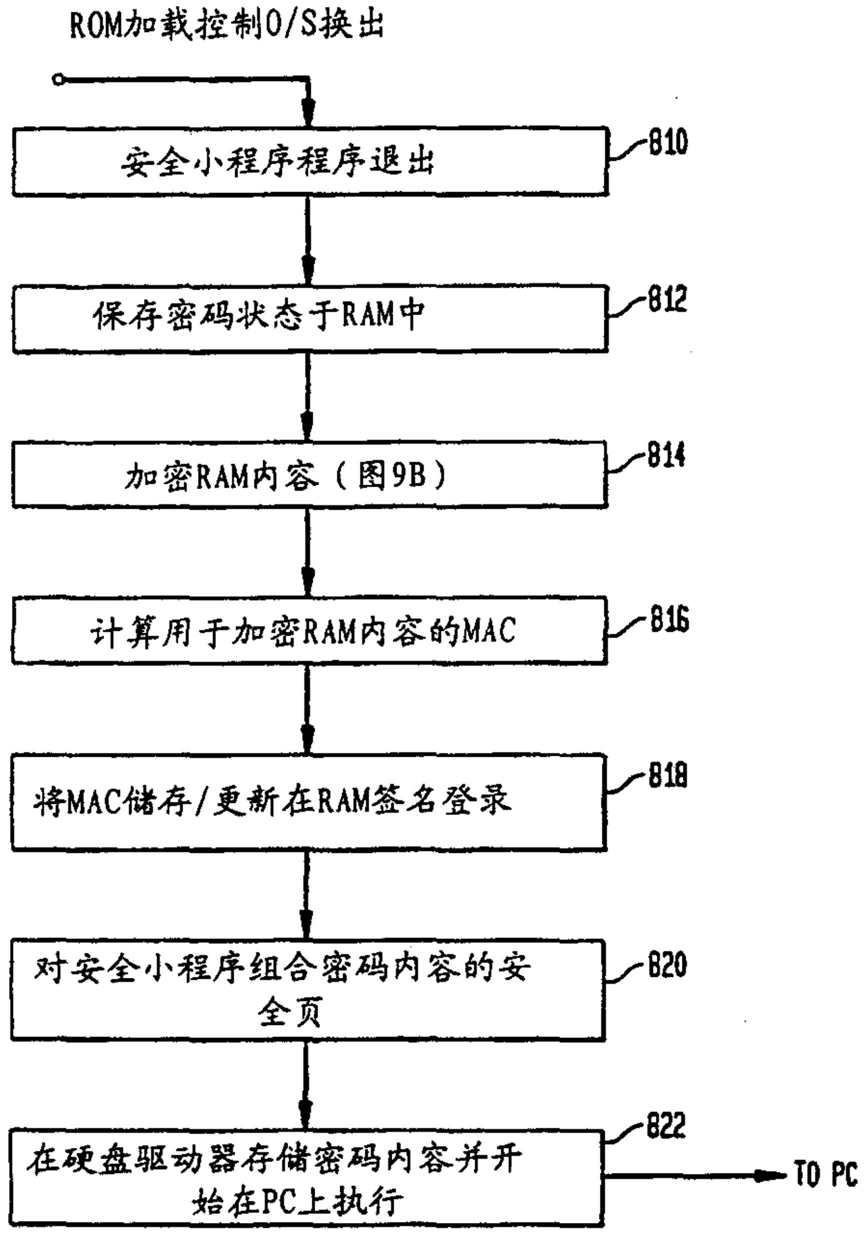
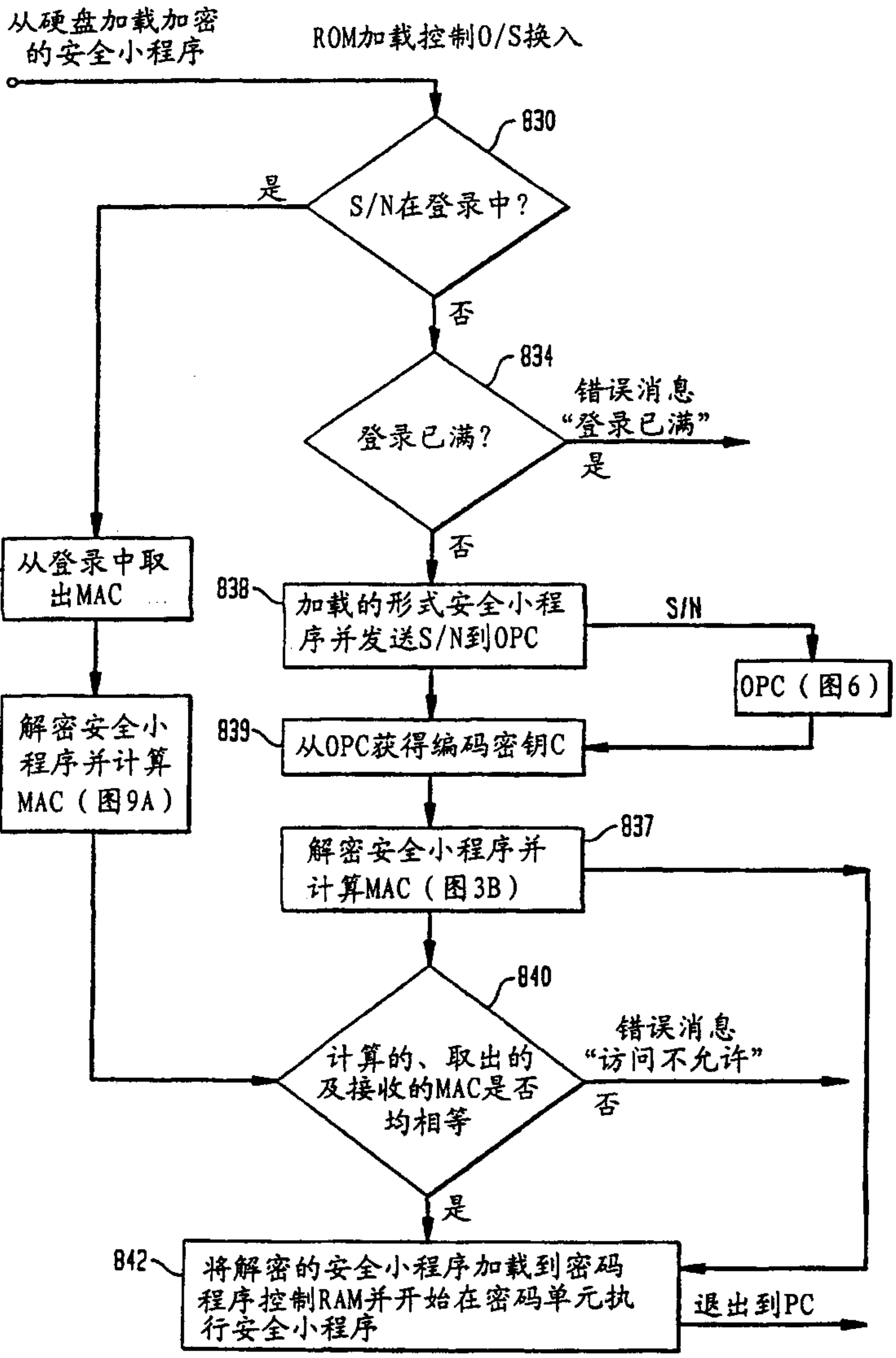

ROM加载控制O/S初始化
PC开机和密码单元初始化
加载自然形式安全小程序
710
712
首次使用密码单元
否
TO PC
是
建立与OPC的通讯并登录密码单元
714
TO PC
OPC
716

图 7



ROM加载控制O/S换出
安全小程序程序退出
810
保存密码状态于RAM中
812
加密RAM内容（图9B）
814
计算用于加密RAM内容的MAC
816
将MAC储存/更新在RAM签名登录
818
对安全小程序组合密码内容的安全页
820
在硬盘驱动器存储密码内容并开始在PC上执行
822
TO PC

图 8A



从硬盘加载加密的安全小程序
ROM加载控制O/S换入
830
S/N在登录中？
是
否
834
登录已满？
错误消息“登录已满”
是
否
832
从登录中取出MAC
838
加载的形式安全小程序并发送S/N到OPC
S/N
OPC（图6）
836
解密安全小程序并计算MAC（图9A）
839
从OPC获得编码密钥C
837
解密安全小程序并计算MAC（图3B）
840
计算的、取出的及接收的MAC是否均相等
错误消息“访问不允许”
否
是
842
将解密的安全小程序加载到密码程序控制RAM并开始在密码单元执行安全小程序
退出到PC

图 8B

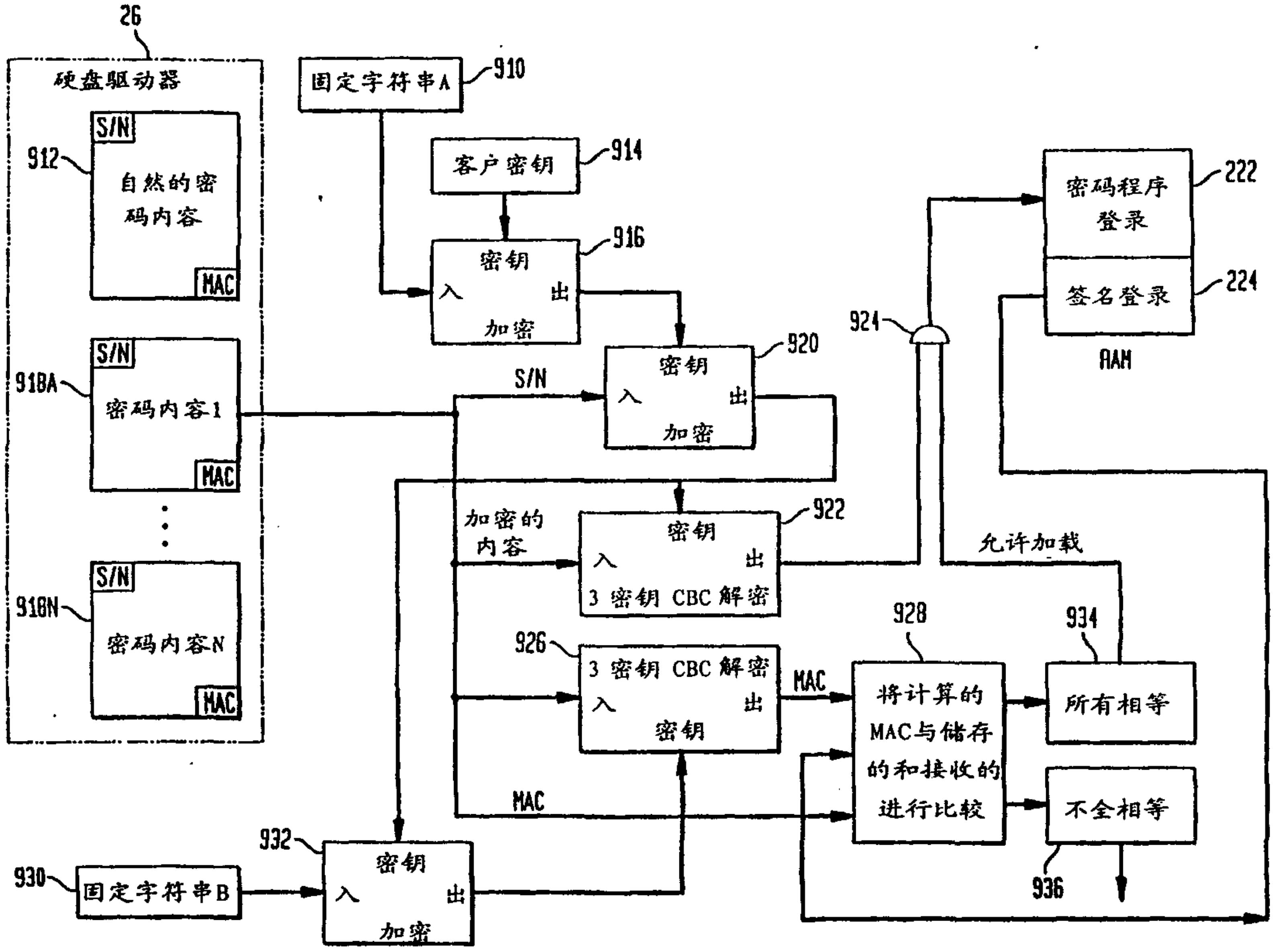
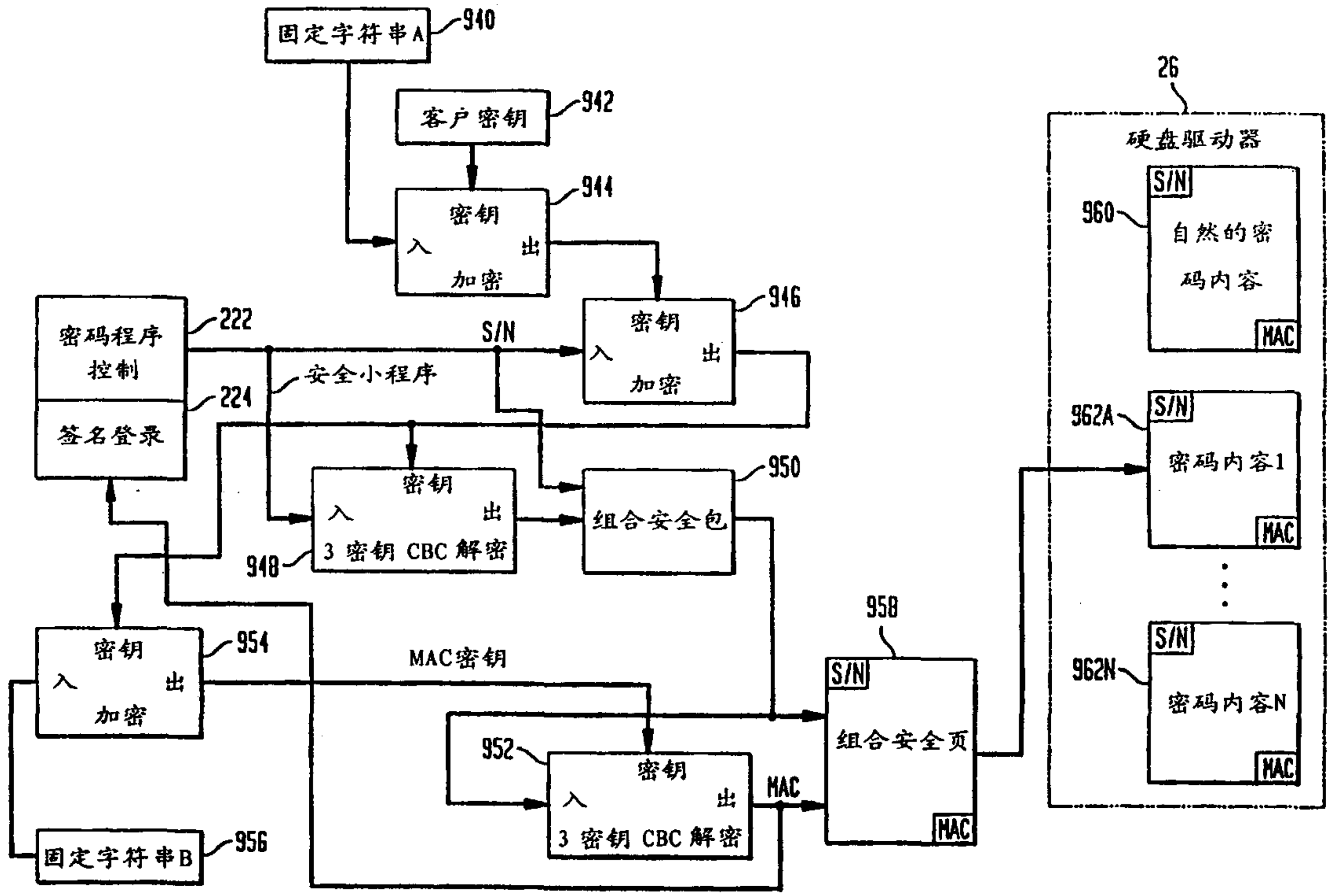

26
硬盘驱动器
S/N
912
自然的密码内容
MAC
918A
密码内容1
918N
密码内容N
固定字符串A
910
客户密钥
914
916
密钥
入
出
加密
920
S/N
922
加密的内容
3 密钥 CBC 解密
926
MAC
924
允许加载
222
密码程序登录
224
签名登录
RAM
928
将计算的MAC与储存的和接收的进行比较
934
所有相等
936
不全相等
930
固定字符串B
932

图 9A

01.11.22



固定字符串A
940
客户密钥
942
密钥
入
出
加密
944
密码程序控制
222
签名登录
224
安全小程序
S/N
密钥
入
出
加密
946
密钥
入
出
3 密钥 CBC 解密
948
组合安全包
950
密钥
入
出
加密
954
MAC密钥
固定字符串B
956
密钥
入
出
3 密钥 CBC 解密
952
MAC
S/N
组合安全页
MAC
958
26
硬盘驱动器
S/N
自然的密码内容
MAC
960
S/N
密码内容1
MAC
962A
S/N
密码内容N
MAC
962N

图 9B