(12) UK Patent Application (19) GB (11) 2 455 796 (13) A

(43) Date of A Publication 24.06.2009

(21) Application No: 0725071.5

(22) Date of Filing: 21.12.2007

(71) Applicant(s):
Nokia Corporation
(Incorporated in Finland)
Keilalahdentie 2-4, 02150 Espoo, Finland

(72) Inventor(s):
Tim Gover

(74) Agent and/or Address for Service:
Symbian Software Limited
2-6 Boundary Row, Southwark, LONDON,
SE1 8HP, United Kingdom

(51) INT CL:
G06F 12/14 (2006.01) ***G06F 21/00*** (2006.01)
G06F 21/24 (2006.01)

(56) Documents Cited:
EP 1678618 A **US 6609199 B1**
US 20020071564 A1

(58) Field of Search:
INT CL **G06F, H04L, H04Q**
Other: **WPI, EPODOC, INSPEC, XPESP, XPIEE, XPI3E**

(54) Abstract Title: Mechanism for controlling access to a key store

(57) A computing device which includes an access control mechanism which is used to control access to keys which are used in cryptographic processes. Any application 303 wishing to gain access to a key from a key store 217 must first obtain authorisation from the access control mechanism. Authorised applications may then access keys directly from the key store, without having to pass data (i.e. the key) through the access control mechanism. The relieves the access control mechanism of some data handling compared with the prior art, which involves the access control mechanism also retrieving the key from the key store and supplying it to the requesting and authorised application(s). The computing device may be a mobile telephone. The application may be identified with a secure ID and have associated capabilities (rights) which the access control mechanism uses to determine if access to the key(s) requested should be granted. The access control mechanism may be a key management server (KMS) 306 in user mode.

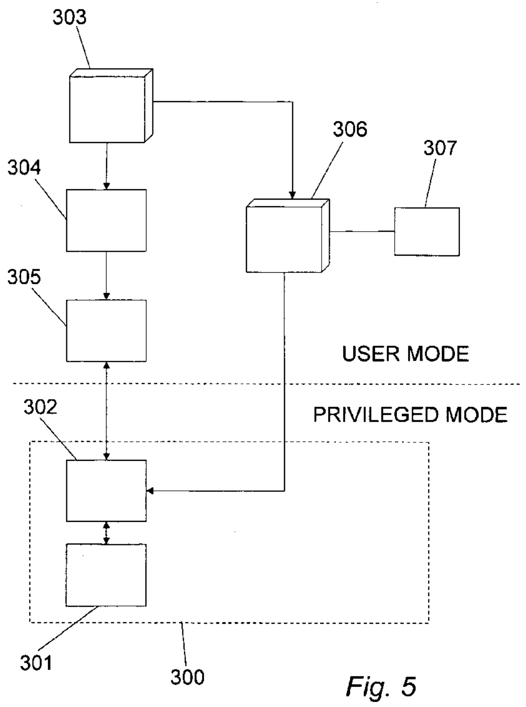


Fig. 5

GB 2 455 796 A

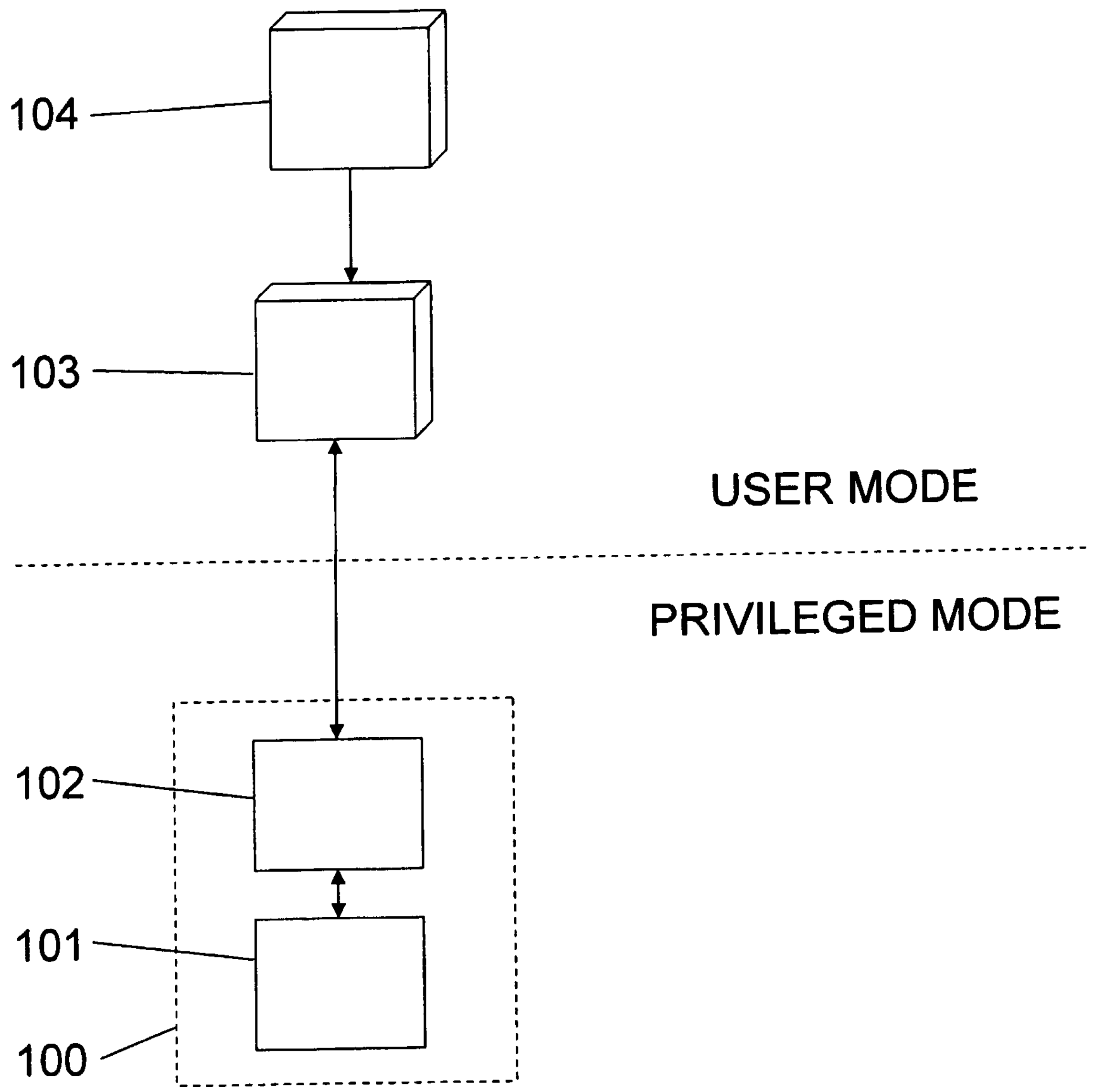


Fig. 1
PRIOR ART

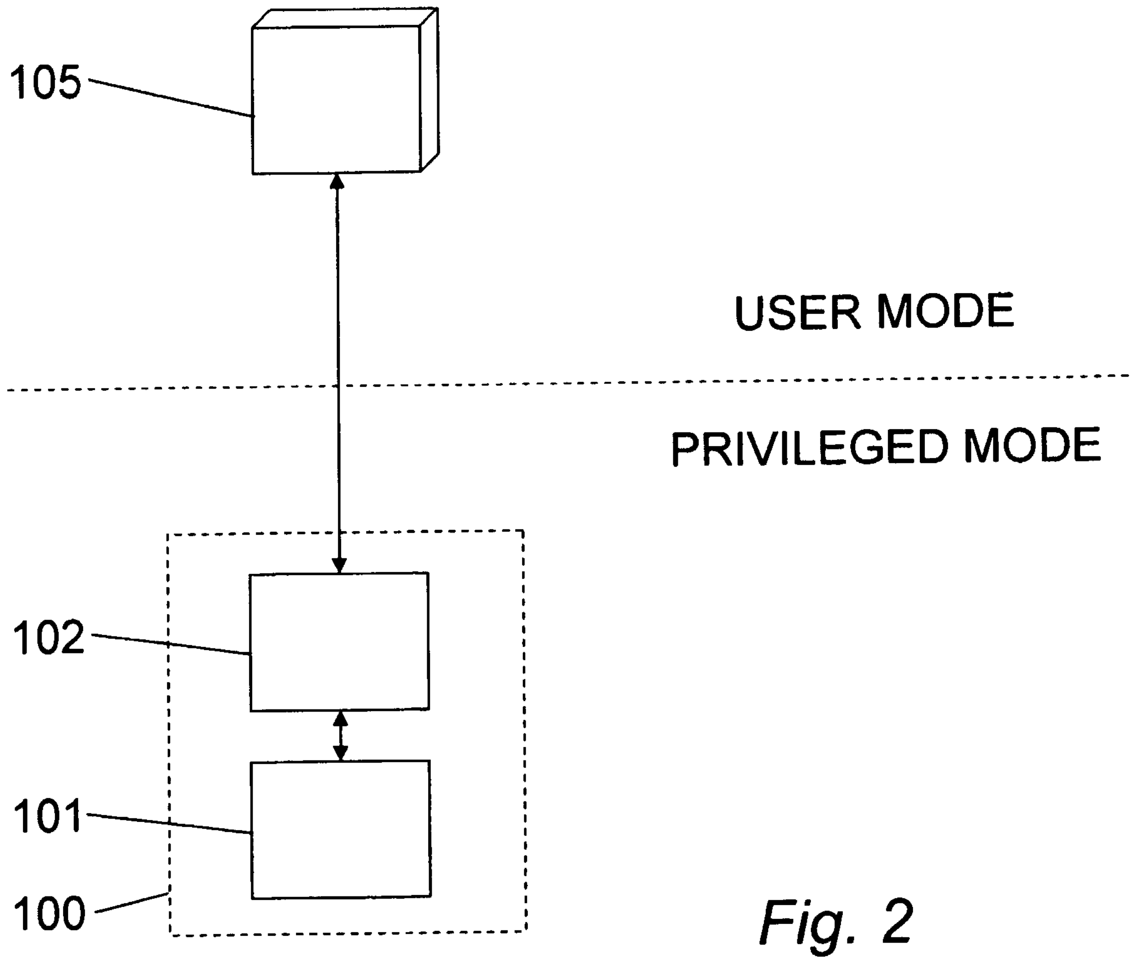


Fig. 2
PRIOR ART

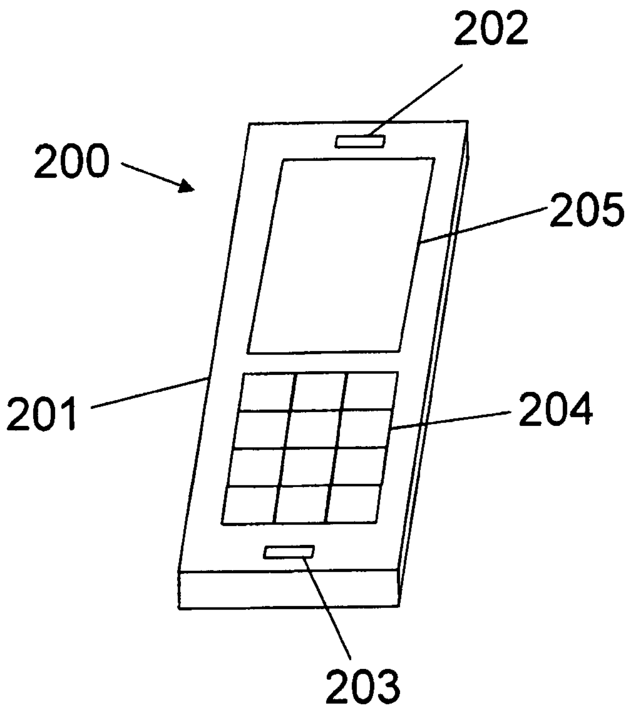


Fig. 3

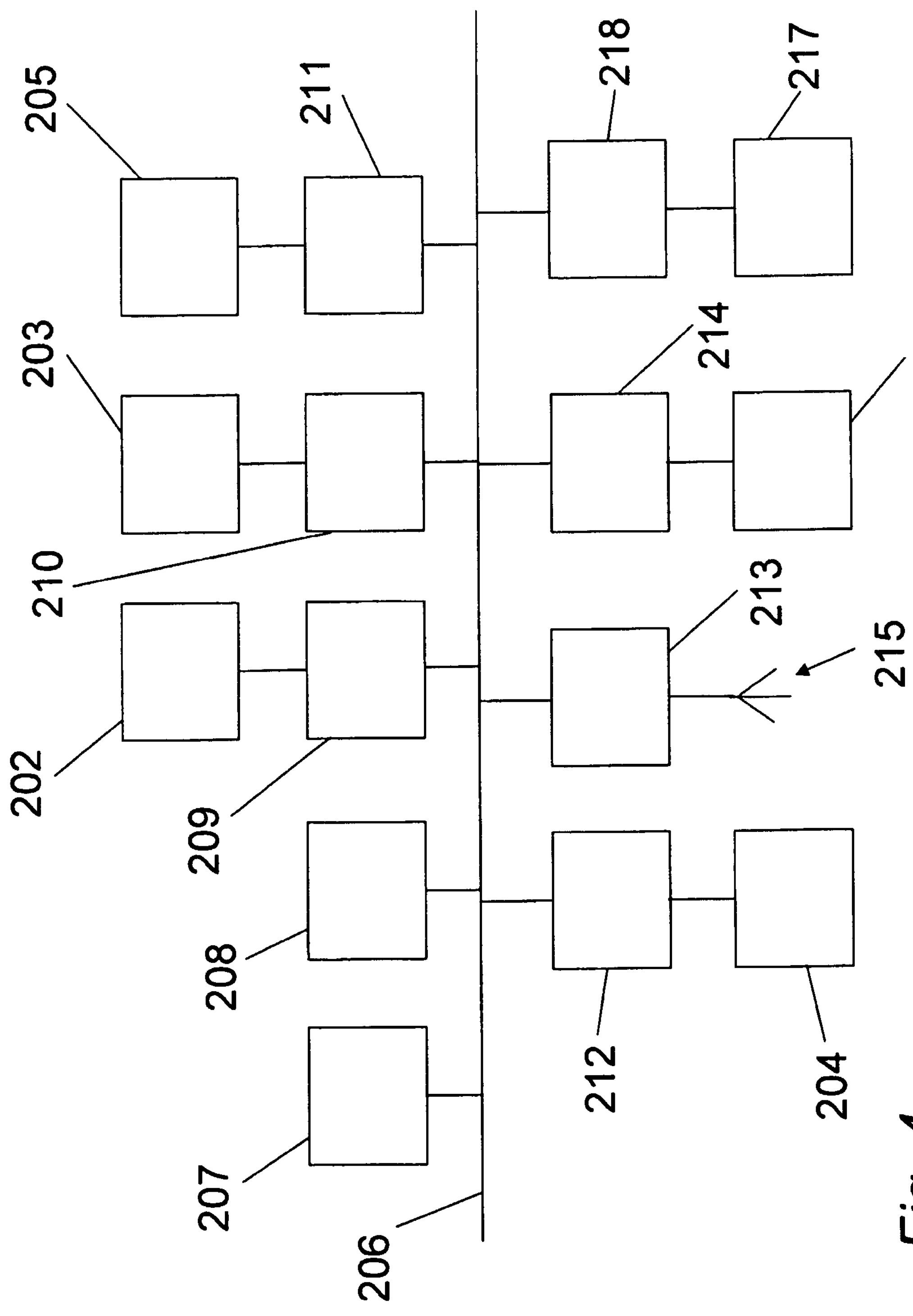

205
211
218
217
203
214
216
210
213
215
202
209
208
207
206
212
204

Fig. 4

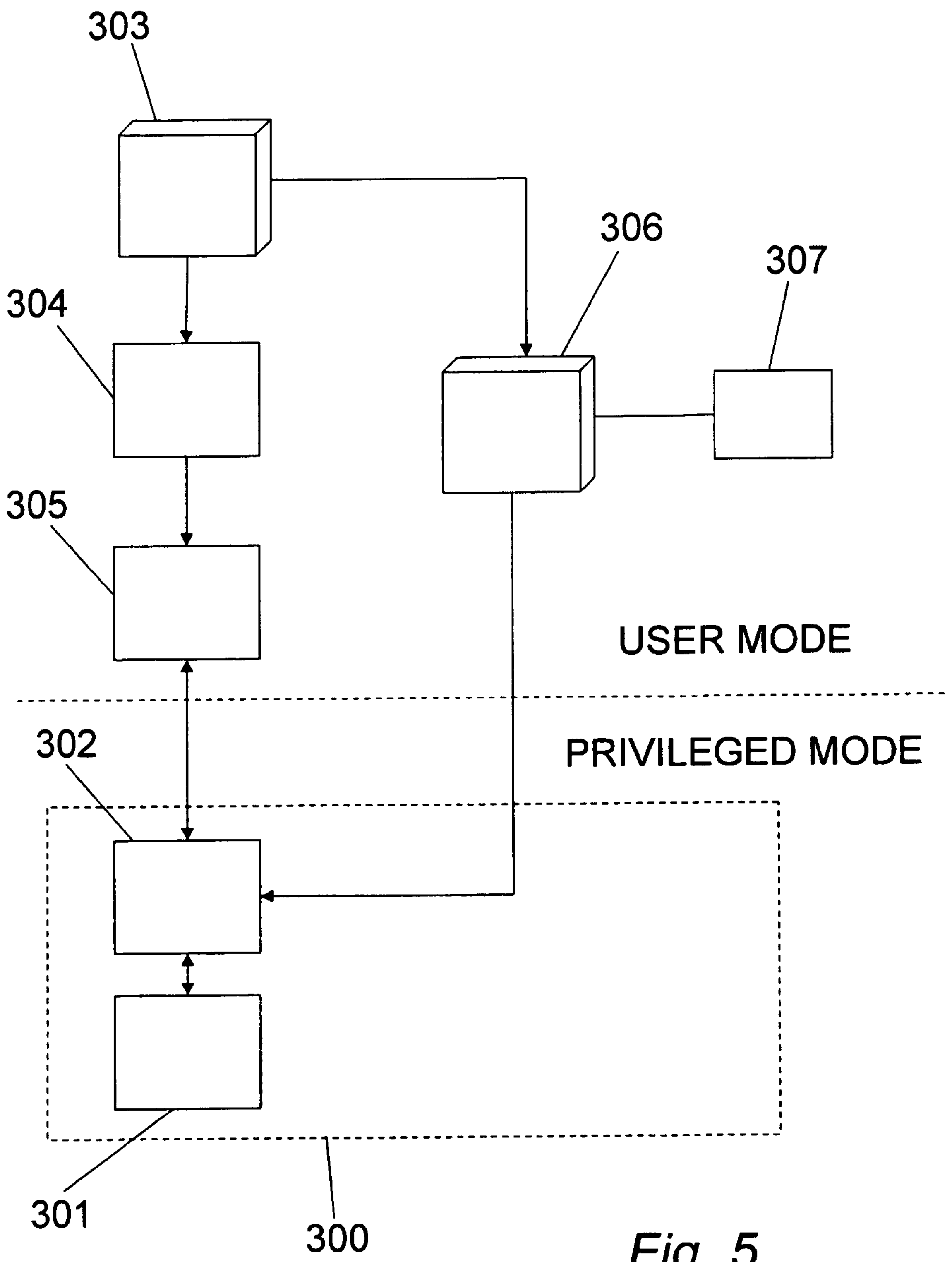
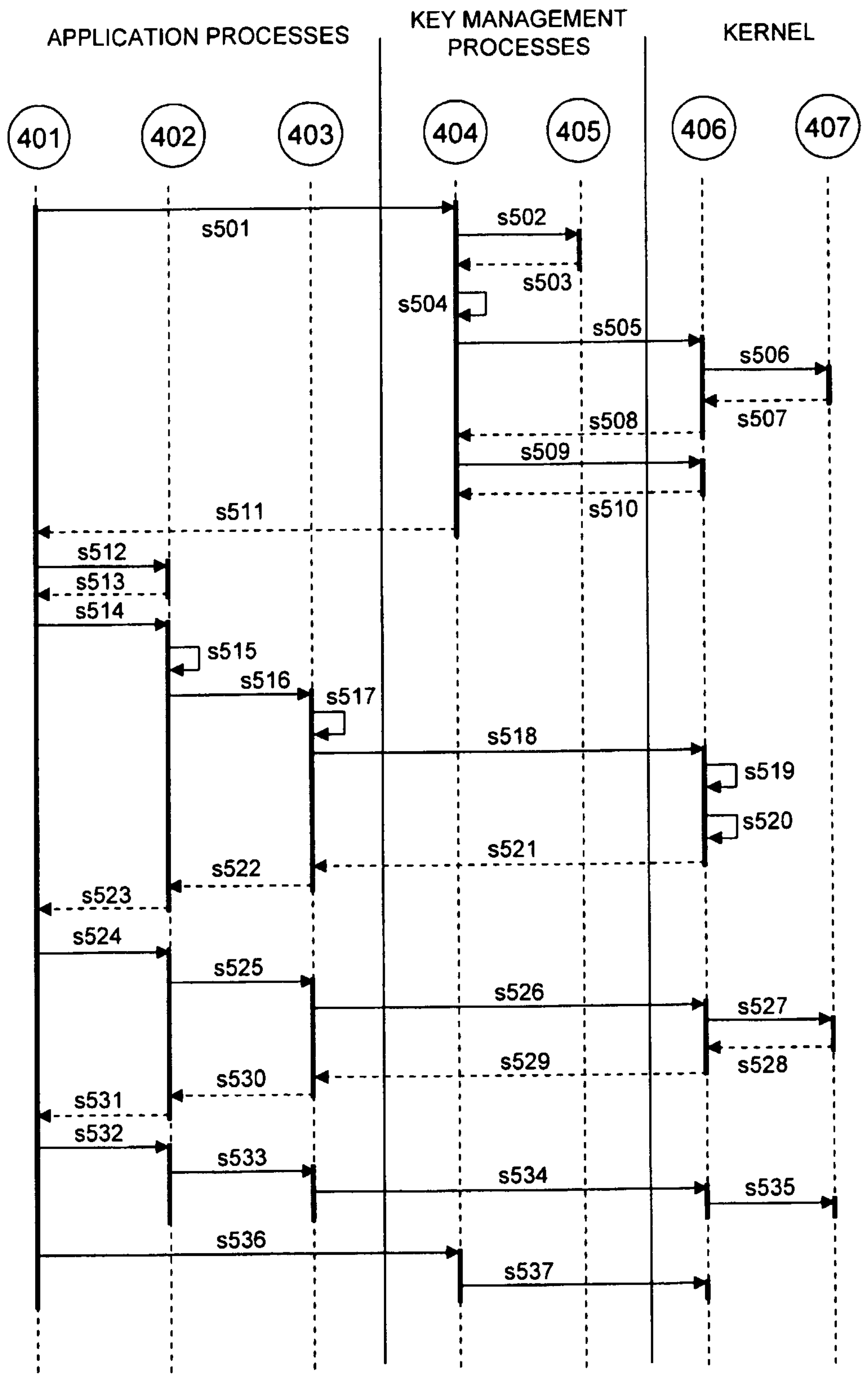


Fig. 5



APPLICATION PROCESSES
KEY MANAGEMENT PROCESSES
KERNEL
401
402
403
404
405
406
407
s501
s502
s503
s504
s505
s506
s507
s508
s509
s510
s511
s512
s513
s514
s515
s516
s517
s518
s519
s520
s521
s522
s523
s524
s525
s526
s527
s528
s529
s530
s531
s532
s533
s534
s535
s536
s537

Fig. 6

Method of access control and corresponding device

The present invention relates to a device including a key store and a mechanism for controlling access to the key store. The present invention also relates to a method for controlling access to a key store.

Background to the Invention and Prior Art

Cryptography is used in a number of different software applications in order to prevent unauthorised third parties gaining access to sensitive or private material. For example, if an email message contains sensitive information, the sender can encrypt the email using an encryption key. Only the authorised recipient, with the correct decryption key, is able to decrypt and view the contents of the email.

Another use of encryption keys is in the field of digital rights management (DRM). Audio files may be encrypted by the vendor of the music so that only those with the correct key can play the music. Typically, anyone purchasing the music legally will be provided with the key necessary for decrypting the audio file.

The keys which are used in cryptographic techniques must be stored safely. In addition, decryption processes can be computationally complex and therefore demand a lot of system processor time. It is therefore known to provide cryptographic acceleration processors, or more generally, cryptographic hardware. Such hardware is used to provide a secure storage area for keys which is separate from the rest of a computer's hardware. In addition, because a processor is dedicated to cryptographic processes, less demand is made of the other resources of the computing device.

In order for any cryptographic system to be secure, access to the keys, and the cryptographic hardware in general, must be controlled. One system which is known from the prior art is shown in Figure 1.

Figure 1 shows a cryptographic device driver 100, which includes a physical device driver (PDD) 101 and a logical device driver (LDD) 102. The cryptographic device driver is typically part of the operating system kernel. The operating system kernel is restricted from access by certain user side applications and is said to run in privileged mode. The cryptographic device driver 100 is the interface between the cryptographic acceleration processor (not shown) and higher level software. The PDD/LDD arrangement will be familiar to those skilled in the art. In the present case, the LDD 102 provides the actual interface to higher layers of the OS.

Also shown in Figure 1 is a dedicated cryptography server 103. This server sits above the kernel in user space and is said to run in user mode. Only the dedicated cryptography server 103 is allowed to talk to the LDD 102. The LDD 102 will not allow any other executable or process to access it. Therefore, any attempt by malicious code to directly access the LDD 102 will fail. Any legitimate application (e.g. application 104) wishing to gain access to a key must do so via the dedicated cryptography server 103. The dedicated cryptography server 103 is therefore responsible for access control. The access control may be achieved using, for example, capabilities, as discussed further below. In order for data to be, for example, decrypted, it must be copied from the application 104 to the dedicated cryptography server 103. Once the data is copied, the dedicated cryptography server 103 is responsible for carrying out the cryptographic operation. Once the result of the operation is determined, the data is copied back to the application 104.

As can be seen, the security policies for the cryptographic system are controlled by the dedicated cryptographic server 103. The main problem with this mechanism is that any data which is to be subjected to cryptographic processing must be copied to and from the dedicated cryptography server 103. This can be a resource-intensive procedure.

Another system known from the prior art is shown in Figure 2. Here, the cryptographic device driver 100 includes the same PDD 102 and LDD 102 as shown in Figure 1. In this system, an application 105 communicates directly with the LDD 102. The cryptographic device driver controls the security policies of the cryptographic system.

The main problem with such a system is that there is generally not enough memory space in the cryptographic hardware for security policies to be stored. Therefore, it is impossible for complex security policies to be implemented with such a system. Also, because the security policies are stored in the cryptographic hardware, modification and expansion of the security policy is more difficult that would be the case with a user space based system, such as the dedicated cryptographic server described above.

Another mechanism for implementing a cryptographic security model is to limit the cryptographic operations to specific processes, such as the media player. Such techniques tend to be used with closed devices where third parties cannot develop and install their own software. Such systems are not flexible at all and the cryptographic operations cannot be extended to other applications.

There is therefore a need for an improved system for managing cryptographic security policies.

Summary of the Invention

The present invention provides a computing device having at least one key stored thereon; wherein control of access to said key is delegated to an access mechanism, stored on the device; and said device is arranged such that said at least one key may be accessed directly if access has been authorised by said access mechanism.

In another aspect, the present invention provides a computing device comprising at least one application, a key store having at least one key, and an access control mechanism, wherein the at least one application is arranged to send requests for key store access to said access control mechanism and to communicate directly with said key store to access said at least one key, the access control mechanism is arranged to receive requests for access to said key store from said at least one application, to determine whether said application is permitted to access said key store and to pass information concerning said at least one application to said key store if said at least one application is authorised to access said key store, and said key store is arranged to receive said information concerning said at least one application from said access control mechanism and to allow access by said at least one application, to said at least one key, if the key store has received information concerning said at least one key from said access control mechanism.

In another aspect, the present invention provides a method of controlling access to at least one key stored on a computing device, the device comprising an access mechanism, the method comprising using the access mechanism to authorise access to said at least one key; and following authorisation, enabling direct access to said key.

Other features of the present invention are defined in the appended claims. Features and advantages associated with the present invention will be apparent from the following description of the preferred embodiments.

Brief Description of the Drawings

The invention will now be described in more detail with reference to the accompanying drawings in which:

Figure 1 shows a cryptographic access system known from the prior art;

Figure 2 shows a further cryptographic access system known from the prior art;

Figure 3 shows a mobile device in accordance with an embodiment of the invention;

Figure 4 is a schematic diagram showing the components of the mobile device of Figure 3;

Figure 5 is a cryptographic access system in accordance with an embodiment of the present invention; and

Figure 6 is an interaction diagram showing process interactions in an embodiment of the present invention.

Description of the Embodiments

Figure 3 shows an example of a device on which the present invention could be implemented. A mobile device 200 comprises an outer casing 201, which includes an earphone 202 and a microphone 203. The mobile device 200 also includes a keypad 204 and a display 205. The keypad 204 enables a user to enter information into the mobile device 200 and instruct the mobile device to perform the various functions which it provides. For example, a user may enter a telephone number, or select another mobile device from a list stored on the mobile device 200, as well as perform functions such as initiating a telephone call.

Figure 4 is a schematic diagram showing the components of the mobile device 200. The device includes a system bus 206 to which the components are connected and which allows the components to communicate with each other. Here, the components are shown to communicate via a single system bus 206, however in practice the mobile device may include several buses to connect the various components. The components of the mobile device 200 include a processor unit 207, memory 208, an earphone controller 209, a microphone controller 210, a display controller 211, a keyboard controller 212, a transceiver 213 and a storage device controller 214. Figure 4 shows a single processor unit 207, however in practice the device may include two or more processor units to control different components of the device. In particular, the device 200 may include a baseband processor unit to control a telephony stack, and an application processor to control an operating system and a user interface of the device. The transceiver 213 is also connected to an antenna 215. The mobile device 200 is arranged to communicate, using transceiver 213, with a base station of a mobile phone network (not shown). The storage device controller 214 is connected to a storage device 216 which may be an internal hard drive or a removable storage device such as a flash memory card.

The mobile device 200 also includes a cryptographic unit 217. The cryptographic unit 217 includes memory (not shown) and a cryptographic acceleration processor (not shown). The cryptographic unit 217 is connected to the system bus 206 by cryptographic unit controller 218. The method of operation of the cryptographic unit 217 will be described below.

Referring to Figure 5, a cryptographic device driver 300 is shown which includes a physical device driver (PDD) 301 and a logical device driver (LDD) 302. The PDD/LDD pair are part of the operating system kernel and therefore operate in privileged mode. The cryptographic device driver 300 provides the interface between the cryptographic unit 217 and higher levels of the operating system. Also shown is an application 303 which is arranged to carry our cryptographic operations. Various information is associated with the application 303 which is unique to the application. For example, the application has a secure ID and may have a number of capabilities. The secure ID is unique to each application and enables the operating system to identify the application. As discussed below, capabilities are the access rights which a particular application has. They allow the operating system to determine what system resources a particular application is allowed to access. When carrying out such an operation, the application loads a dynamic linked library (DLL) known as a cryptographic SPI (system program interface). In use, the cryptographic SPI DLL 304 is loaded into application 303 and runs as an application process. When an application requires use of a key stored in the cryptographic unit 217, the cryptographic SPI DLL 304 loads hardware plug-in DLL 305. As with the cryptographic SPI DLL 304, in use, the hardware plug-in DLL 305 is loaded into the application 303 and is run as an application process.

Capabilities allow an operating system to carefully control which resources can be accessed by which executables and by which static or dynamic linked libraries. Executables may be assigned a set of capabilities which specify which resources the executable is allowed to access and what actions that executable is allowed to carry out. In an operating system which uses servers to control access to particular resources, the servers determine whether an executable is allowed access to a particular resource, based on its capabilities.

The hardware plug-in DLL 305 enables the application 303 to communicate directly with the LDD 302. However, this does not necessarily mean that the application 303 will be granted access to the cryptographic unit 217, as will be described below. The cryptographic system shown in Figure 5 also includes a key management server (KMS) 306. The KMS 306 is arranged to control access by applications to the cryptographic unit 207. The KMS 306 is directly accessible by application 303. The KMS 306 is also able to communicate with the LDD 302. Also shown is a key management database (KMD) 307. The KMD 307 includes information regarding which applications are allowed to access which keys.

The application 303 has capabilities associated with it as noted above. In the present case, the application 303 may have a capability which indicates that the application is allowed to access a particular key. The KMS 306 checks the application's capabilities and will allow access to the key, as described below, if the application has the correct capability.

The operation of the cryptographic system will now be described in more detail.

Figure 6 is an interaction diagram which shows the interactions between application processes, key management processes and kernel processes. The application processes include application 303 processes 401, cryptographic SPI DLL 304 processes 402 and hardware plug-in DLL 305 processes 403. As noted above, the cryptographic SPI DLL 304 and the hardware plug-in DLL 305 are both DLLs which are loaded by the application 303. All of these executables are therefore run as part of the application processes. The key management processes include KMS 306 processes 404 and KMD 307 processes 405. Finally, the kernel includes the cryptographic device driver processes. These are LDD 302 processes 406 and PDD 301 processes 407.

The process of carrying out a key operation begins with application 303 determining the identity of the key which it requires. The application 303 sends a request for access to the specified key to the KMS 306 (s501). The request includes the secure ID and the capabilities for the application 303. The KMS 306 then checks the KMD 307 (s502) in order to determine whether or not the application 303 is allowed access to the specified key. The KMD 307 contains meta-data which includes information defining key usage and key ownership information. The KMD 307 then returns the relevant meta-data to the KMS 306 (s503). The KMS 306 uses the secure ID and capabilities of the application 303 to determine whether or not the application 303 is allowed access (s504).

One of the main benefits of this arrangement is that the security policies implemented by the KMS 306 can be as extensive and as complex as the device manufacturer desires. There is no limit on the size of the KMD 307 or on the size of the KMS 306 because they both reside in user space. This is in contrast to the prior art shown in Figure 2. Here, because the security policies are implemented in the cryptographic device driver 100, there is a clear limit on the amount of memory which can be assigned to security policy code and meta-data.

Assuming the application 303 is allowed access to the specified key, the KMS 306 sends a request to the LDD 302 (s505) to open a handle to the specified key. The KMS 306 also has a secure ID. The LDD 302 knows the secure ID of the KMS 306 and always allows the KMS 306 access. The LDD 302 checks with the PDD 301 to ensure that the requested key exists in the cryptographic unit (s506). In turn, the PDD checks the hardware key store to ensure that the key exists. Assuming the specified key does exist, the PDD 301 informs the LDD 30 that this is the case (s507). The LDD 302 then passes the specified key handle back to the KMS 306 (s508). The KMS 306 sends the secure ID of the application 303 to the LDD 302 together with details of the specified key to which the application 303 should be allowed access (s509). The KMS 306 also informs the LDD 302 of the operations which the application is allowed to perform with the specified key. For example, if the application 303 has requested the key for the purposes of decryption, the application will not be allowed to use the key for other cryptographic purposes. The LDD 302 the sends a confirmation that the secure ID of application 303 has been registered (s510). The KMS 306 stores the specified key handle in a key information object and passes a handle to that object back to the application 303 (s511). This is the last part of the process in which the KMS 306 takes part. All future process interactions are between application processes and kernel processes.

In order for the application to carry out the necessary key process, it must first load the cryptographic SPI DLL 304. Once loaded, the cryptographic SPI DLL 304 brings together the necessary elements for the key process to be carried out. The cryptographic SPI DLL 304 includes a generic key factory. The generic key factory is arranged to load the necessary cryptographic algorithm and produce a generic key object. The data from the application and the key from the cryptographic unit are ultimately loaded into the generic object so that the key process can be carried out. Once the cryptographic SPI DLL 304 is loaded, the application 303 calls the cryptographic SPI DLL 304 (s512). The application indicates that the key is a non-extractable key located in the cryptographic unit 217 and passes the identity of the specified key together with the identity of the required algorithm. The cryptographic SPI DLL then returns the generic key object to the application 303 (s513). The application 303 then sends a request to the cryptographic SPI DLL 304 (s514) which then. looks-up the necessary plug-in for the specified key and algorithm (s515). The cryptographic SPI DLL 304 then loads the hardware plug-in DLL 305 (s516). The hardware plug-in DLL 305 checks to see if it recognises the specified key handle (s517). The hardware plug-in DLL 305 then attempts to open a handle to the LDD 302 (s518). The LDD 302 checks the secure ID of the caller against the secure IDs which have been registered for the specified key (s519). In the present case, the secure IDs match and the LDD creates a key handle (s520). The key handle is passed back to the hardware plug-in DLL 305 (s521), which passes the key handle to the cryptographic SPI DLL 304 (s522) which passes the key handle to the application 303 (s523).

Once the application 303 has the key handle, it can obtain the key from the LDD 302. The application 303 sends an instruction to the LDD 302 (s526) to create a key object via the cryptographic SPI DLL 304 (s524) and the hardware plug-in DLL 305 (s525). The LDD 302 then obtains the specified key from the PDD 301 (s527 + s528) and returns the key in a key object to the hardware plug-in DLL 305 (s529). The hardware plug-in DLL 305 passes the key object to the cryptographic SPI DLL 304 (s530) which passes the key object to the application 303 (s531). The application can then carry out the necessary key process using the specified key.

Once the cryptographic operation is complete, the application 303 closes the cryptographic SPI DLL 304 (s532) which in turn closes the hardware plug-in DLL 305 (s533). The handle to the key is also closed (s534 + s535) The application the informs the KMS 306 that it has finished with the specified key (s536 + s537). The KMS 306 then informs the LDD 302 that the application has finished with the key and the secure ID for the application 303 is unregistered by the LDD 302.

As a result of the above described processes, data which is used in the cryptographic process can be loaded into memory, by said application 303, and modified in the application memory space. The data does not need to be copied to another memory space in order for the cryptographic operation to be carried out. This is in contrast to the prior art shown in Figure 1. Here, all data which is to be subjected to cryptographic processing must be copied to the dedicated cryptographic server. This increases the burden on device resources and slows the whole process down. The above described embodiment does not suffer from the disadvantages of data copying which the prior art suffers from.

In the above embodiment, the client application is always trying to use a key stored in the key store. If an application provides its own key, it is of course possible for such an application to perform any operation it requires with that key.

It will be appreciated by the skilled reader that the above embodiment uses two access lists. One access list is a persistent access list which is kept by the KMD 307. The other list is the non-persistent temporary list kept by the LDD 302. The LDD only needs to store information relating to the keys which are currently in use. This information can be stored in kernel side device memory together with other normal data. The persistent access list is kept in user space by the KMS 306. This file may be encrypted and be protected by a specific hardware key known to the KMS 306.

In the above description the cryptographic mechanism has been described as allowing direct access to the key store. In the context of the present application, "directly" is intended to mean that, once authorisation is confirmed, applications may access the key store without involving the key management server. The term "directly" should not be taken to mean that the application is not required to communicate with the key store through other layers of code. For example, the application may be required to access the key store via layers of the operating system which include the device drivers. This is "direct" access in the context of the present invention.

In the above description, various processes including decryption and signing have been described. The term cryptographic encompasses these processes as well as any other processes which require the use of keys or similar devices which are used in processes of this nature.

The above described embodiment relates to keys which are stored in dedicated cryptographic hardware. The present invention may also be implemented using keys which are protected by a software based key store. For example, the keys may be stored in the device main memory and be protected by an operating system based key store. The access control mechanism of the present invention may also be applied to such an environment.

The term "client" is intended to refer to any executable, library or other entity which may require access to a particular key. It is not intended to be limited to particular types of entities such as user applications. It is intended to include any entity that requires services of another entity.

Various modifications, changes, and/or alterations may be made to the above described embodiments to provide further embodiments which use the underlying inventive concept, falling within the spirit and/or scope of the invention. Any such further embodiments are intended to be encompassed by the appended claims.

Claims

1. A computing device having at least one key stored thereon; wherein control of access to said key is delegated to an access mechanism, stored on the device; and said device is arranged such that said at least one key may be accessed directly if access has been authorised by said access mechanism.

2. A computing device according to claim 1, further comprising at least one client, arranged for use with said at least one key, wherein said client is arranged to access said at least one key directly, if access is authorised by said access mechanism.

3. A computing device according to claim 2, wherein said at least one key is arranged for use in at least one cryptographic process and said device is arranged such that when access has been authorised by said access mechanism, access to said key, for use in said cryptographic process, is not through said access mechanism.

4. A computing device according to claims 3, further comprising data, for use with said at least one key in said at least one cryptographic process; wherein said device is arranged such that, during operation of said cryptographic process, said data is not copied to said access mechanism.

5. A computing device according to claim 4, wherein said at least one client is at least one application, wherein said at least one application is arranged to load said data into a memory space of said device and further arranged such that said data is modified by said cryptographic process in said memory space.

6. A computing device according to claim 5, wherein said at least one application is arranged to, first, inform said access mechanism that access to

said at least one key is required and, second, access said at least one key directly.

7. A computing device according to claims 5 or 6, wherein said at least one application has at least one capability and said access mechanism uses said at least one capability to determine if said at least one application is allowed to access said at least one key.

8. A computing device according to claim 7, wherein said at least one application has an associated unique identity which said access mechanism and said key store use to identify the at least one application.

9. A computing device according to any of claims 5 to 8, further comprising at least one library, wherein said at least one application is arranged to load a said at least one library prior to accessing said at least one key.

10. A computing device according to any preceding claim, further comprising a key store in which said at least one key is stored.

11. A computing device according to claim 10 when dependent on any of claims 5 to 9, wherein said access mechanism is arranged to inform said key store that said at least one application is authorised to access said at least one key.

12. A computing device according to claim 11 wherein said key store is arranged to be accessed directly, if access has been authorised by said access mechanism.

13. A computing device according to any of claims 10, 11 or 12, wherein said access mechanism operates independently of said key store.

14. A computing device according to any of claims 10, 11, 12 or 13, further comprising a cryptographic hardware unit which includes said key store and a cryptographic acceleration processor.

15. A computing device according to claim 14, further comprising a cryptographic device driver, the device driver arranged to provide an interface between the cryptographic hardware unit and said at least one application and said access mechanism.

16. A computing device according to any preceding claim wherein said access mechanism is a key management server.

17. A computing device according to claim 16, wherein said key management server operates in user mode.

18. A computing device according to any preceding claim, wherein the computing device is a mobile telephone.

19. A computing device comprising: at least one application; a key store having at least one key, and an access control mechanism, wherein the at least one application is arranged to send requests for key store access to said access control mechanism and to communicate directly with said key store to access said at least one key; the access control mechanism is arranged to receive requests for access to said key store from said at least one application, to determine whether said application is permitted to access said key store and to pass information concerning said at least one application to said key store if said at least one application is authorised to access said key store; and said key store is arranged to receive said information concerning said at least one application from said access control mechanism and to allow access by said at least one application, to said at least one key, if the key store has received information concerning said at least one key from said access control mechanism.

20. A method of controlling access to at least one key stored on a computing device, the device comprising an access mechanism, the method comprising: using the access mechanism to authorise access to said at least one key; and following authorisation, enabling direct access to said key.

21. A method according to claim 20, wherein said device further comprises at least one client, and said step of enabling direct access is a step of enabling said client to directly access said at one least key.

22. A method according to claim 21, wherein said step of direct access is carried out independently of said access mechanism.

23. A method according to claim 22, wherein said device further comprises data, for use with said at least one key; wherein said method does not include the step of copying said data to said access mechanism.

24. A method according to any claim 23, further comprising the step of using said data and said at least one key in a cryptographic process.

25. A method according to claim 24, wherein said at least one client is at least one application, the method further comprising using said at least one application to load said data into a memory space of said device, wherein said step of using said data in said cryptographic process includes the step of modifying said data in said memory space.

26. A method according to any of claims 20 to 25, further comprising the steps of informing said access mechanism that access to said at least one key is required and accessing said at least one key directly.

27. A method according to claim 26, wherein said device further comprises a key store in which said at least one key is stored and the method further comprises informing said key store that access to said at least one key is authorised.

28. A method according to claim 27, wherein said step of enabling direct access is a step of enabling direct access to said key store.

29. A method according to claim 25, wherein said at least one application has at least one capability, said method further comprising using said at least one capability to determine if said at least one application is allowed to access said at least one key.

30. A method according to any of claims 20 to 29, wherein said access mechanism operates in user mode.

31. A method according to any of claims 20 to 30, wherein the computing device is a mobile telephone.

32. A computer program or suite of computer programs arranged such that when executed by a computer they cause the computer to operate in accordance with the method of any of claims 20 to 31.

33. A computer readable medium storing the computer program, or at least one of the suite of computer programs, according to claim 32.

34. An operating system for causing a computing device to operate in accordance with a method as claimed in any one of claims 20 to 31.

Intellectual Property Office

For Creativity and Innovation

Application No:	GB0725071.5	**Examiner:**	Mr Adam Tucker
Claims searched:	1-34	**Date of search:**	4 April 2008

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular relevance
X	1-7, 10, 12, 13, 18, 20-26 & 28-34	EP 1678618 A (RESEARCH IN MOTION) See the whole document
A	-	US 2002/0071564 A1 (Kurn et al.) See in particular the abstract and paragraphs 50, 62-64, 75 & 115-119
A	-	US 6609199 B1 (DeTreville) See, for example, claims 1-4

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art
Y	Document indicating lack of inventive step if combined with one or more other documents of same category	P	Document published on or after the declared priority date but before the filing date of this invention
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC^X :

Worldwide search of patent documents classified in the following areas of the IPC

G06F; H04L; H04Q

The following online and other databases have been used in the preparation of this search report

WPI, EPODOC, INSPEC, XPESP, XPIEE, XPI3E

International Classification:

Subclass	Subgroup	Valid From
G06F	0012/14	01/01/2006
G06F	0021/00	01/01/2006
G06F	0021/24	01/01/2006