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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 January 2017 (19.01.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/011051 A2

(51) International Patent Classification:
H04L 9/18 (2006.01) *G06F 21/72* (2013.01)

(21) International Application Number:
PCT/US2016/029144

(22) International Filing Date:
25 April 2016 (25.04.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/153,671 28 April 2015 (28.04.2015) US

(71) Applicant: **SEQUITUR LABS, INC.** [US/US]; 22525 SE 64th Place, Suite 294, Issaquah, Washington 98027 (US).

(72) Inventors: **ATTFIELD, Philip**; 4740 285th Avenue SE, Fall City, Washington 98024 (US). **DOYLE, Michael**; 10234 Spider Rock Avenue, Las Vegas, Nevada 89135 (US). **TING, Vincent**; 1475 W King Edwards Avenue, Vancouver, British Columbia V6H 2A3 (CA).

(74) Agent: **LARIVIERE, F. David**; LaRiviere, Grubman PC, 200 Sky Park Drive, Monterey, California 93940 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) Title: SECURE DATA PROTECTION AND ENCRYPTION TECHNIQUES FOR COMPUTING DEVICES AND INFORMATION STORAGE

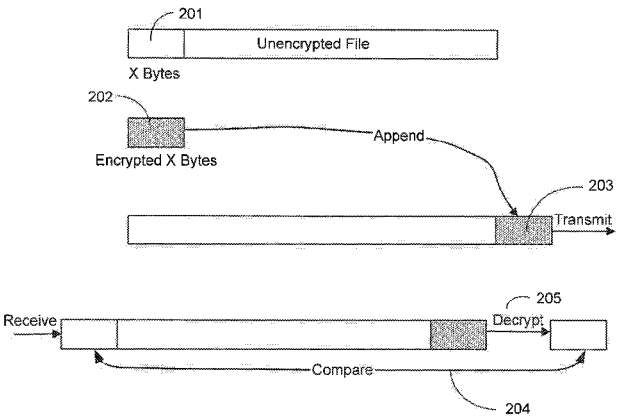


Figure 2

(57) Abstract: A system for secure data protection and encryption for computing devices. The present invention includes a fast encryption technique for quickly ensuring that the correct binding parameters are used for an encrypted data file. The encrypted file is used in two ways. Because unsecure data could pass through a peripheral device to gain access to a secure computing environment, a dongle housing encryption and decryption subsystems is placed in between the unsecure sources and the peripheral that can encrypt and decrypt data intended for the secure computing environment. The firmware of the computing device can be updated by dividing the update file into encrypted segments that are verified on the device and placed into non-volatile memory. When all parts have been received, decrypted, and written into memory, the device reboots using the updated firmware.

WO 2017/011051 A2

UTILITY PATENT APPLICATION

SECURE DATA PROTECTION AND ENCRYPTION TECHNIQUES FOR COMPUTING DEVICES AND INFORMATION STORAGE

INVENTOR: PHILIP ATTFIELD, MICHAEL DOYLE, AND VINCENT TING

PRIORITY CLAIM

[0000] This non-provisional application claims priority to Provisional Patent Application Serial No. 62/153,671, entitled "Secure Data Protection and Encryption Techniques for Computing Devices and Information Storage" filed April 28, 2015.

TECHNICAL FIELD

[0001] The present invention relates to the technical fields of Computer Security, Embedded Systems, Encryption, Mobile Computing, Telecommunications, Digital Communications, and Computer Technology.

BACKGROUND OF THE INVENTION

[0002] Recent years have brought the emergence and rapid proliferation of mobile computing devices such as mobile telephones or "handsets" with extensive computing, communication, and input and interaction capabilities ("smart phones") plus a growing array of other mobile computing devices such as touchscreen tablets, "netbooks", electronic

document readers, and laptops in a wide range of sizes and with wireless and wired communication capabilities.

[0003] This proliferation of mobile devices has been accompanied by complementary advances in development and adoption of long range wireless broadband technologies such as 3G and 4G, as well as commonplace deployment of shorter range wireless technologies such as the 802.11 series of wireless standards and "Bluetooth" short range wireless, all with considerable bandwidth. These technologies span multiple radio frequency bands and protocols.

[0004] Alongside the radio transceivers for such communications capabilities, many of these mobile devices also contain an array of onboard sensors such as cameras, microphones, and GPS receivers plus other locating technologies, as well as considerable fixed-onboard and removable memory for information and multimedia storage.

[0005] Furthermore, smartphones and similar devices are typically capable of running a wide variety of software applications such as browsers, e-mail clients, media players, and other applications, which in some cases may be installed by the user. Mobile devices are now fully capable computing environments that require security to the same extent as any other computing environment would.

[0006] Along with the profusion of smartphones and other mobile, wireless-capable devices, there has also been a dramatic increase in the use of social networks and related technologies for information sharing for consumer as well as for professional uses. Access to social networks on mobile devices has heightened concerns about individual, government, and corporate information security, and about possibilities for privacy

violations and other unintended and undesirable information sharing. Furthermore, the possible professional and personal use of any given handset presents a complex set of usage contexts under which rules for device capability usage and information access need be considered.

[0007] Such sophisticated and capable smartphones and similar devices, along with the vast amounts of information that they can contain and access, present a large set of potential security vulnerabilities (a large "attack surface") that might allow information to be accessed by malicious parties or allow undesirable use and exploitation of the device capabilities for malicious purposes such as "phishing" fraud, other online fraud, or inclusion in botnets for spam transmission, denial-of-service attacks, malicious code distribution, and other undesirable activities.

[0008] Furthermore, compared with conventional desktop personal computers, smartphone handsets by nature are portable and thus more easily stolen. Portability also means that the devices will encounter security contexts that cannot be foreseen, and which may never occur again. The mobile threat landscape is complex and presents a vast set of extant and emergent security concerns.

[0009] There is, therefore, a growing need to improve upon not only the degree of protection provided by components and systems that enhance the security of mobile devices, but also to improve on the efficiency and security of such security-related components and systems themselves, so that both they and the devices and the information that they protect are more robust and are better able to withstand attempts to thwart or otherwise compromise them.

BRIEF SUMMARY OF THE INVENTION

[0010] Because modern mobile systems must connect to external (peripheral) devices and they must also be able to update their firmware and other executable programs in a secure manner, the present invention is an efficient system and method to encrypt files for transmission, and then either pass them securely through peripheral devices or directly to the firmware memory for updating. There are three key steps presented. The encryption is made more efficient by appending a small data segment to the end of the encrypted file to ensure that the binding parameters being used are correct. Peripherals are connected to the system via a dongle for encrypting or decrypting files that pass through the peripheral device that sits between the secure and normal operating environments. Finally, the invention proposes a system and method to securely update firmware or other executable programs to secure memory by segmenting the update file into signed and encrypted parts that are transmitted separately. They are then received, decrypted, and written to a secure memory location. When complete the system reboots (restarts) with the new updated firmware or executable program.

[0011] The invention has a number of advantages. The encryption method is more efficient than conventional methods because it is able to ensure that the best binding parameters are used. The dongle then uses the encryption technique to ensure that the inherent vulnerability of unsecure data reaching the secure world via peripheral devices is closed. All data is now encrypted. Lastly, the encryption technique is applied to each data segment of the firmware update or executable program to be written to ensure that the data that reaches the secure memory location is verified.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Figure 1. Schematic representation of system.

[0013] Figure 2. The process of fast encryption and decryption for secure binding.

[0014] Figure 3. The system for securing the pathway between the secure and normal world by placing a dongle in between the peripherals and the host.

[0015] Figure 4. The process for secure firmware updates by segmenting the update, encrypting, transmitting, decrypting, and writing to a secure memory location.

DETAILED DESCRIPTION OF THE INVENTION

[0016] The present invention consists of a set of capabilities and techniques for enhancing the security and privacy of information storage on computing devices, and for performing secure updates to network-connected computing devices.

Encryption Techniques for Secure Binding and Fast Decryption Processing

[0017] Security of encryption of digital data for mobile devices can be enhanced by binding the encryption key to one or more specific digital objects, such that it is difficult or impossible to decrypt the data without the presence of, and access to, those objects. As a non-limiting example, such binding may be performed through the use of some complex computation with inputs that include each of the digital objects to be bound to, and with one of the outputs being the encryption key. In the following list, a set of such objects is presented which may be used in some combination to perform such secure binding:

- Program or module binding.
- Program version or module version binding.
- Developer certificate binding

- Device binding. In the case of device binding, the computed key is based on a parameter or on parameters unique to the device, such as the International Mobile Station Equipment Identity (IMEI), such that the encrypted file can only be decrypted on the same device.
- Custom: allows the use of a custom parameter as an input to which to bind the key.
- Password: allows the use of a password as an input to which to bind the key.

[0018] The set of objects used for binding can be a combination of these listed objects, as selected prior to or during binding.

[0019] Among the drawbacks to encryption, and particularly to public key encryption over symmetric encryption, are the computational time and computational expense required for decryption of the encrypted data (Figure 2). Furthermore, in some cases, it may not be known with certainty that the binding parameters about to be used for decryption are the correct ones. Steps can be taken during encryption to allow for faster processing when the file is later decrypted. As one such step, a small segment of a certain initial segment of the unencrypted file (201) is copied, pre-processed and encrypted (202), then finally appended to the end of the encrypted file (203). That segment, rather than the whole file, can then be examined post-encryption to determine whether the binding parameters to be used in decryption are the correct ones (204).

[0020] As a specific, non-limiting example, during the encryption process, the Secure Hash Algorithm SHA256 is applied to a copy of the first X bytes (201) of the unencrypted file and the encrypted copy is appended to the whole file, encrypted (202), to the end of the encrypted file (203). After this procedure, post-encryption, unencrypt and extract the

appended SHA256 component (205) and compare it with the first X bytes (204), and verify that they match, as a fast test that the binding parameters being used in the decryption are correct.

Methods for Securing Peripherals

[0021] One key approach to defending security-related systems, data, and components from malicious attack is to have them reside within especially secure areas, partitions, or environments on device hardware that are inaccessible to unauthorized parties or for unauthorized purposes, with this "secure world" separated from the main device operating system and, in some cases, from certain of its insecure resources, with these insecure components comprising the "normal world". A further degree of security can be provided if the secure partitions or areas are also invisible and undetectable to the greatest degrees possible, under unauthorized circumstances and by unauthorized parties.

[0022] However, even with the use of such secure areas and other measures internal to the computer, due to a need for interaction and input with computer users in many cases, weak points for security of entire systems remain, such as the input and output devices themselves, and their connections and interfaces with other system components and with operating system software and applications. Malicious use of software for intercepting keyboard entries, or "key logging", is common, as is "phishing" software for unwanted interception of entered data, and this represents a considerable threat that can reveal passwords and other sensitive data to parties not intended to see it.

[0023] In the present invention (Figure 3), the following means provide for more secure coupling of input and output devices to secure system components and applications on a

host computer. Here, secure transmission of data to or from a peripheral device is provided through a "normal" or untrusted partition or channel, from or to a secure partition or secure area on the host, and with the data securely protected during its passage (301). Non-limiting examples of such peripherals include keyboards, keypads, trackpads, touchscreens, mice, camera, biometric sensors, active display devices such as LCD displays and monitors, printers, and plotters. In order to do this, a hardware device hereinafter referred to as "dongle", containing certain required components is inserted or installed between the peripheral and the host (302).

[0024] Connections between peripheral, dongle, and host may be through standard interfaces such as USB, PS/2, or by other means, but in each case with the dongle securely separating peripheral and host. The components include storage for digital encryption key(s) as needed for encryption of data prior to transmission, as well as required hardware for interfacing with the input device and the computer. This allows for data to be encrypted before transmission to the normal world of the host for passage through to the secure world for use in secure or otherwise trusted applications there such as payment applications or healthcare-related applications. Corresponding encryption keys are stored in the secure world (303) for appropriate encryption data as needed for communication with the secured peripherals. Each secured peripheral has a corresponding "handler" module (304) in the secure world for managing communication out into the normal world through to the peripheral.

[0025] In some cases, peripheral input and output cannot be communicated directly to a secure world, and in such cases, a normal world application may be required as an

intermediary to route traffic from the input device such as a keyboard, to a trusted application for processing, but in such a case the data is already encrypted prior to reaching the intermediary. Similarly, output to a display device could be done using an intermediary normal world application driving a GPU, but with though the normal world to a display being encrypted by an intermediary dongle.

[0026] In cases where such secure communication with the peripheral is not required, the dongle is permitted to go into a non-encrypting mode and act as a transparent "pass through" of unencrypted or otherwise unsecured data to a normal world unsecured application, upon receipt of an acceptable signal such as a key sequence or special key code from the user.

Methods for Secure Device Firmware Updates

[0027] Modern computing devices such as smartphones, tablet computers, and also internet-connected devices in the "Internet of Things", typically contain non-volatile memory and persistent memory content collectively known as "firmware" that is stored in the non-volatile memory on the device. Firmware content may include operating system code, "boot loader" code for device initialization and loading the operating system, and other code essential to device operations, plus data and in some cases application software.

[0028] Inevitably, whether to fix software bugs, patch security vulnerabilities, to update features, or for other reasons, it is desirable to be able to update the firmware content. For reasons such as lower cost, manageability, and ease of update, it can be preferable to update the firmware via the internet or some other network connection, rather than by other means such as hardware replacement or by transfer of the new firmware content from a locally

connected storage device. However, such "Over the Air" (OTA) network updates present certain security risks such as possible malicious interception of the in-transit firmware and subsequent injection of malicious code onto the device (401), or other undesired exploits of firmware target content (402). The following methods are presented for securing such firmware updates (Figure 4).

[0029] First, the method requires that at least some amount of size U of non-volatile memory (403) be designated for use for the secure updating process. The new firmware of the update, the net "payload" in this case, is encrypted that can be securely verified on the target device, and split into data segments of size U or smaller (404). Each segment is digitally signed. A secure other portion of non-volatile memory such as a locked "boot" portion must contain trusted drivers (405) for network connections such as wired internet or wireless technology such as 802.11 variants or cellular, and/or other network modalities, and that portion or another locked portion should contain certificates or root certificates as required to decrypt received segments of encrypted payload. The means to call cryptographic verification functions should also either reside in or have its driver in the boot sector (406).

[0030] Data segments are then transmitted over the network (407) from a secure server or other secure source and received via the trusted network driver (405) and used to update other non-volatile memory areas appropriately. Data segments may be received sequentially and loaded into successive non-volatile memory locations in received order, or in other cases may be loaded in any sequence and may be loaded into assigned non-volatile memory locations designated for individual identified segments; these and other segment

transmission possibilities are contemplated within the invention. As each data segment is received, its signature is verified (406), and the data segment is written to the appropriate portion of non-volatile memory (408). The download, verify, install procedure of data segments is repeated until the entire set of segments, comprising the update payload is received. Once entire set of data segments has been installed into memory, the device can boot (restart) with the newly updated firmware. Note that this procedure is not exclusive to firmware updates but can be used to safely write any executable program into secure memory.

[0031] This method has the strengths or advantages as follows:

1) The network driver used for OTA updates is trustworthy regardless of device state. This is important because if the device is in a partially-updated state, then a trustworthy loader is required throughout uploading for confidence of fully secure update and for secure recovery to be possible in case of failed transmission of any chunks or in case of any other update problems.
2) A reliable fallback mechanism exists in that a partially-updated device can still boot with network connectivity because the driver and ability to check payload reside in a trusted area.

INDUSTRIAL APPLICATION

[0032] The invention is generally useful to any application where the computing device used may be connected to external peripheral devices and must be updated periodically. Neither the firmware update technique nor the peripheral dongles require the use of the aforementioned encryption technique but some accepted form of encryption must be

employed. Any computing device can use these techniques, not only mobile or handheld devices.

CLAIMS

What is claimed is:

1. A system for secure data protection for computing devices each having non-volatile secure memory and that communicate with peripheral devices configured to accept data from unsecure sources and to transmit data to the non-volatile secure memory comprising:

   a fast encryption subsystem for efficiently encrypting data;

   a fast decryption subsystem for efficiently decrypting data from the fast encryption subsystem and for ensuring that the correct binding parameters are used;

   a dongle for housing a peripheral security subsystem having the fast encryption subsystem and the fast decryption subsystem for encrypting and decrypting data that passes through the peripheral devices to and from, respectively, the non-volatile secure memory.

2. A system for secure data protection and updating for computing devices each having non-volatile secure memory and that communicate with peripheral devices configured to that accept update data from unsecure sources and to transmit update data to the non-volatile secure memory comprising:

   a fast encryption subsystem for efficiently encrypting update data, said update data being data that includes an executable program;

   a fast decryption subsystem for efficiently decrypting update data from the fast encryption subsystem and for ensuring that the correct binding parameters are used;

   a dongle for housing a peripheral security subsystem having the fast encryption subsystem and the fast decryption subsystem for encrypting and decrypting update data that passes through the peripheral devices to and from, respectively, the non-volatile secure memory; and

a secure updating subsystem for securely updating software executable programs stored in the non-volatile secure memory on the computing device.

3. The system of claim 1, wherein the fast encryption subsystem includes:

a segmentation subsystem for dividing the unencrypted data into separate data segments each of which are less than a preselected byte size;

a binding subsystem for binding the encryption key to an object;

an encryption subsystem for encrypting all of the data segments and for appending a copy of the first encrypted segment to the end of the series of encrypted segments; and

a transmission subsystem for transmitting the encrypted data segments with the appended data segment to the non-volatile secure memory.

4. The system of claim 1 wherein the fast decryption subsystem includes:

a receiver subsystem for receiving encrypted data segments from the fast encryption subsystem;

a trusted network driver coupled to the receiver subsystem for ensuring that all the encrypted data segments are received from a trusted source and for recomposing the data segments in original order;

an extraction subsystem for extracting and decrypting the last data segment and for comparing it to the decrypted first data segment for ensuring that the correct binding parameters are used;

a verifier for decrypting the encrypted data segments and for verifying the authenticity of the data segments and for writing the decrypted data segments into the non-volatile secure memory.

5. The system of claim 3 wherein the encryption subsystem digitally signs each of the encrypted data segments.

6. The system of claim 3, wherein the object that the encryption key is bound to includes at least one of:

a program module;

a program version;

a developer certificate;

a device;

a password; and

a custom binding defined by the user.

7. The system of claim 1, wherein the peripheral security subsystem includes:

a dongle for each peripheral device having a fast encryption subsystem and a fast decryption subsystem housed therein for encrypting and decrypting data that passes through peripheral devices to and from the non-volatile secure memory; and

a handler within the non-volatile secure memory associated with each peripheral device for managing communication to and from the unsecure environment into the non-volatile secure memory via that peripheral device.

8. The system of claim 2, wherein the secure updating subsystem includes:

a network interface for reading unencrypted update data from an external source;

a segmentation subsystem for dividing the unencrypted update data into separate unencrypted update data segments each of which are less than a preselected byte size;

an encryption subsystem for encrypting all of the update data segments and for appending a copy of the encrypted first update data segment to the end of the series of encrypted update data segments;

a transmitter for transmitting the encrypted update data segments to the non-volatile secure memory via the network;

a trusted network driver for ensuring that the encrypted update data segments are received from a trusted source;

a verifier for decrypting the encrypted update data segments, for comparing the decrypted last update data segment to the first decrypted update data segment to ensure that the correct binding parameters are used, for verifying the authenticity of the update data segments, and for writing the decrypted update data segments into the non-volatile secure memory in their original order; and

a restart subsystem for restarting the software executable program within the non-volatile secure memory.

9. The system of claim 8, wherein the encryption subsystem further digitally signs each update data segment.

10. A method for secure data protection for computing devices each having non-volatile secure memory and that communicate directly with peripheral devices that accept data from unsecure sources and transmit data to the non-volatile secure memory comprising the steps of:

inserting a dongle having a fast encryption subsystem and a fast decryption subsystem housed therein between each peripheral device and each unsecure source for fast encrypting and decrypting of data;

segmenting the unencrypted data from the unsecure source into separate unencrypted data segments each of which are less than a preselected byte size and encrypting each data segment;

transmitting each encrypted data segment to the non-volatile secure memory;

decrypting each encrypted data segment within the non-volatile secure memory; and

recomposing the decrypted data by sequencing the data segments in original order.

11. A method for secure data protection and updating for computing devices each having non-volatile secure memory and that communicate directly with peripheral devices that accept update data from unsecure sources and transmit update data to the non-volatile secure memory comprising the steps of:

inserting a dongle having a fast encryption subsystem and a fast decryption subsystem housed therein between each peripheral device and each unsecure source for fast encrypting and decrypting of update data;

segmenting the unencrypted update data from the unsecure source into separate unencrypted update data segments each of which are less than a preselected byte size and encrypting each update data segment;

transmitting each encrypted update data segment to the non-volatile secure memory;

decrypting each encrypted update data segment within the non-volatile secure memory;

recomposing the decrypted update data by sequencing the update data segments in original order; and

restarting the executable program with the new update after all update data segments have been received and recomposed.

12. The method of claim 10 wherein the fast encryption of data includes the steps of:

binding an encryption key to an object;

creating a copy of an unencrypted data segment of a preselected byte size from the beginning of the data to be encrypted;

encrypting the copy of the unencrypted data segment using an encryption process;

encrypting all of the other unencrypted data segments;

appending the encrypted copy of the first data segment to the end of the encrypted data segments; and

transmitting the complete encrypted file.

13. The method of claim10 wherein the fast decryption of data includes the steps of:

receiving the encrypted file;

extracting the appended encrypted copy of the first data segment from the end of the encrypted file;

decrypting the encrypted copy of the first data segment;

comparing the decrypted copy of the first data segment with the unencrypted first data segment from the beginning of the encrypted file for identifying the correct binding parameters; and

decrypting all of the other encrypted data segments using the correct binding parameters.

14. The method of claim 10 wherein secure data protection for computing devices further includes the steps of:

intercepting data from the non-volatile secure memory intended for use in an unsecure computing environment using the dongle; and

decrypting the data using the dongle for use in the unsecure computing environment.

15. The method of claim 10 wherein secure data protection for computing devices further includes the steps of:

intercepting data from an unsecure computing environment intended for use in the non-volatile secure memory using the dongle;

encrypting the data using the dongle for use in the non-volatile secure memory; and

verifying the data passed into the non-volatile secure memory with a handler for decrypting data.

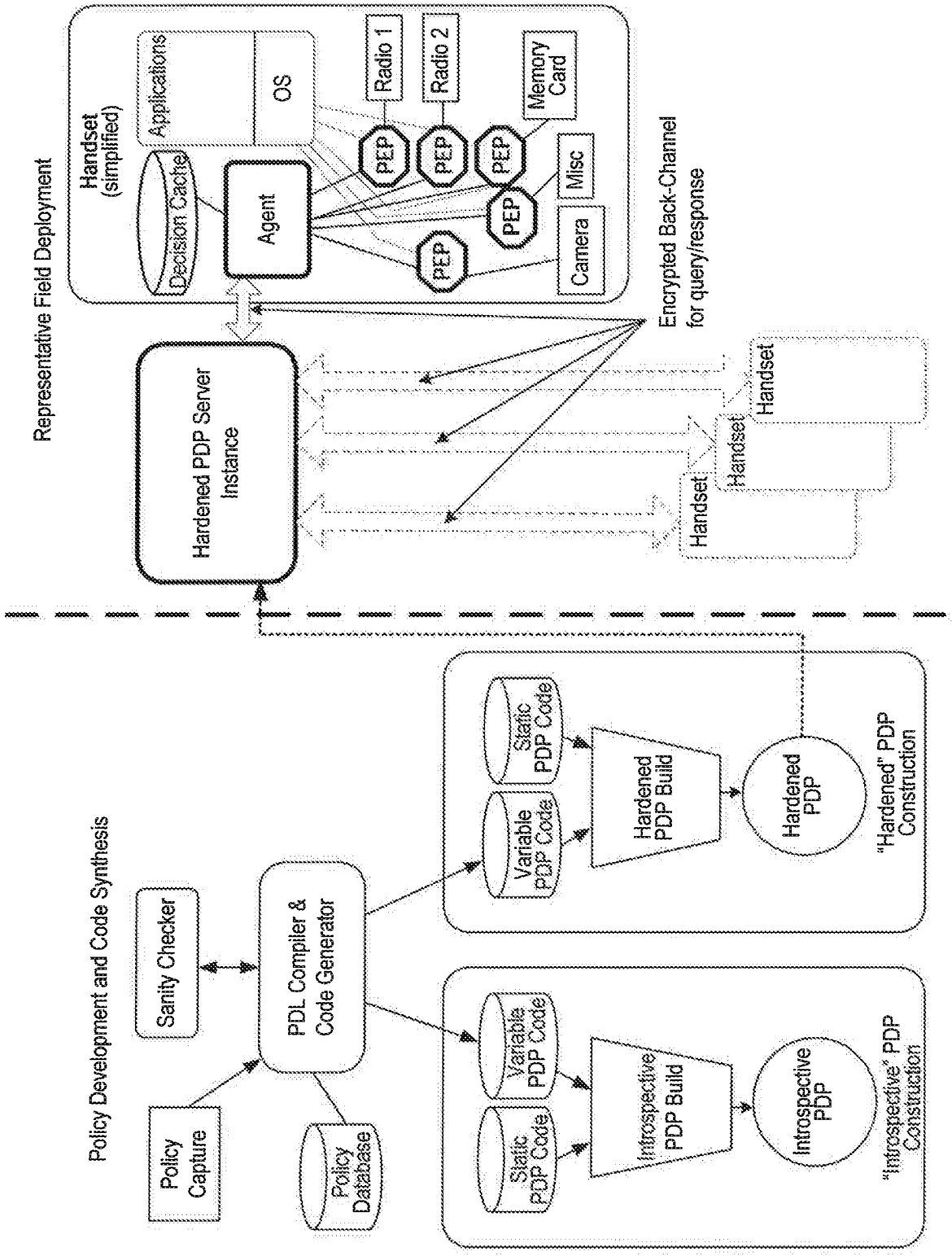

Representative Field Deployment
Handset (simplified)
Applications
OS
Decision Cache
Agent
Radio 1
Radio 2
Memory Card
PEP
Misc
Camera
Hardened PDP Server Instance
Encrypted Back-Channel for query/response
Handset
Policy Development and Code Synthesis
Sanity Checker
Policy Capture
PDL Compiler & Code Generator
Policy Database
Static PDP Code
Variable PDP Code
Introspective PDP Build
Introspective PDP
"Introspective" PDP Construction
Hardened PDP Build
Hardened PDP
"Hardened" PDP Construction

Figure 1

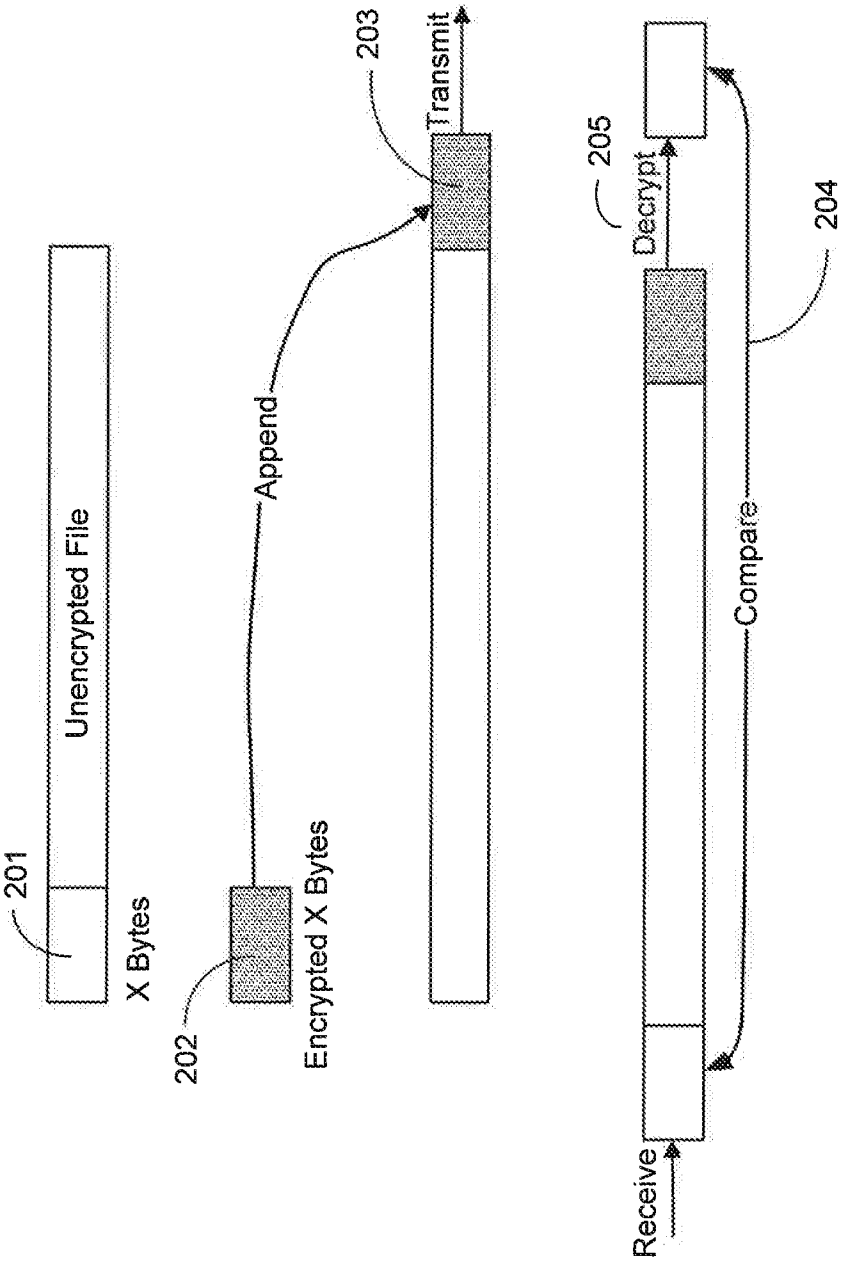

Unencrypted File
201
X Bytes
202
Encrypted X Bytes
Append
203
Transmit
205
Decrypt
204
Compare
Receive

Figure 2

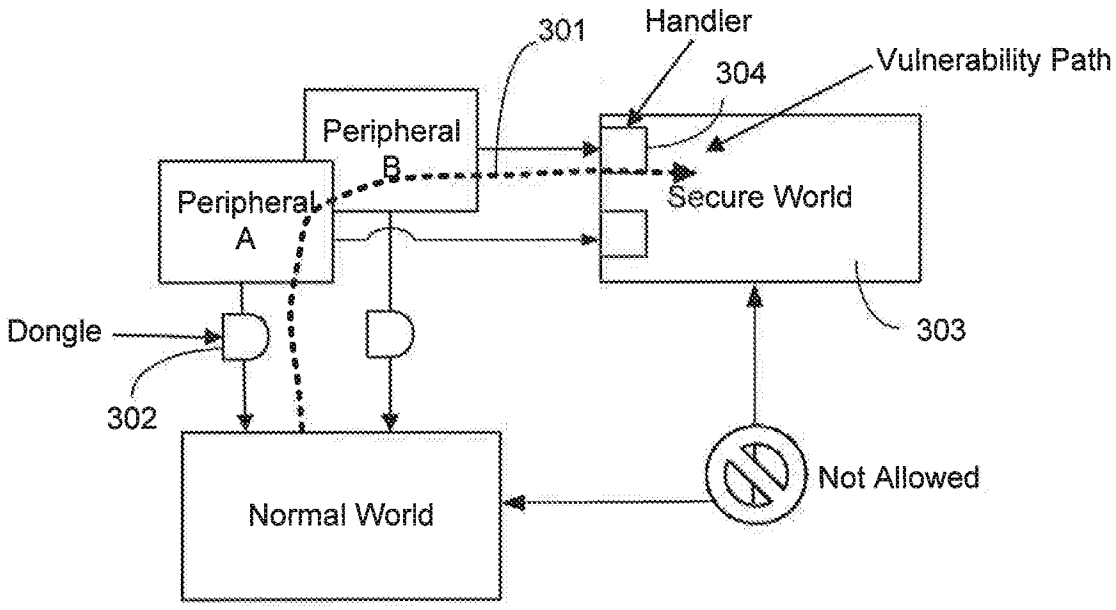

301
Handler
304
Vulnerability Path
Peripheral B
Peripheral A
Secure World
Dongle
303
302
Normal World
Not Allowed

Figure 3

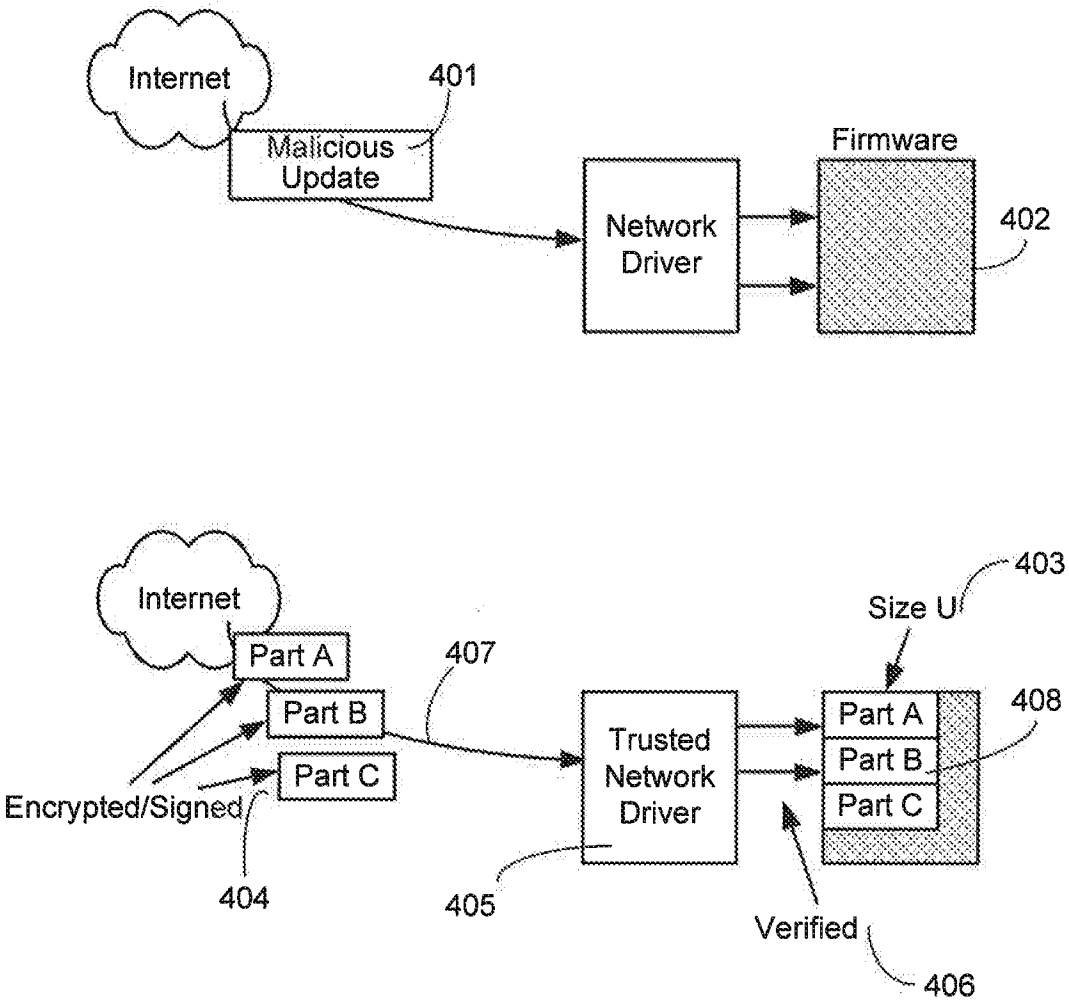

Internet
401
Malicious Update
Network Driver
Firmware
402
Internet
Part A
Part B
Part C
Encrypted/Signed
404
407
Trusted Network Driver
405
Size U
403
Part A
Part B
Part C
408
Verified
406

Figure 4