



(51) International Patent Classification:

G06F 21/44 (2013.01) G06F 21/51 (2013.01)  
G06F 21/12 (2013.01)

(21) International Application Number:

PCT/US2016/030136

(22) International Filing Date:

29 April 2016 (29.04.2016)

(25) Filing Language:

English

(26) Publication Language:

English

(71) Applicant: HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).

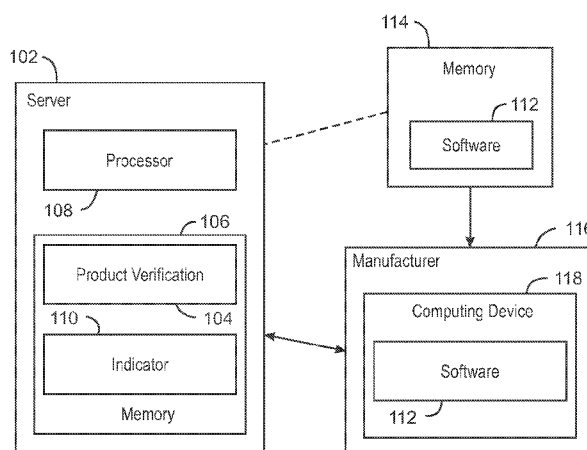
(72) Inventors: OH, Sung; 8000 Foothills Blvd., Roseville, California 95747 (US). GOODWIN, Barry L.; 8000 Foothills Blvd., Roseville, California 95747 (US).

(74) Agent: ORTEGA, Arthur S. et al.; Hewlett Packard Enterprise, Mail Stop 79, 3404 E. Harmony Road, Fort Collins, Colorado 80528 (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).

(54) Title: PRODUCT VERIFICATION



100  
FIG. 1

(57) Abstract: In examples, techniques are disclosed for product verification of a computing device including calculating a first indicator as a function of software module prior to the software module being stored on the computing device, calculating a second indicator as a function of the software module as stored on the computing device, and comparing the second indicator to the first indicator.

**Declarations under Rule 4.17:**

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

**Published:**

- *with international search report (Art. 21(3))*

## PRODUCT VERIFICATION

### **BACKGROUND**

[0001] During the manufacture and/or shipping of computing devices, the software (e.g., firmware, embedded software, etc.) on the computing device may be compromised such as altered by unauthorized entities or individuals. Indeed, the software may be unscrupulously amended, and also undesired or unauthorized software added to the computing device in addition to the software intended by the supplier, and so forth. The product received by the customer may be compromised. Unfortunately, detection of compromise of a computing device may be problematic. A need exists to improved detection of compromise of a computing device product that contains firmware and/or embedded software and has been compromised during the manufacturing process or while in route to the customer, for example. Conventional solutions to provide physical security control of product packaging such as tamper proof tape may not address the manufacturing process or logistics operations.

### **BRIEF DESCRIPTION OF THE DRAWINGS**

[0002] Certain exemplary embodiments are described in the following detailed description and in reference to the drawings, in which:

[0003] FIG. 1 is a block diagram of system having a server for product verification in relation to a manufacturer in accordance with examples;

[0004] FIG. 2 is a block diagram of system having a server for product verification in relation to a customer site in accordance with examples;

[0005] FIG. 3 is a block diagram of a method of product verification in relation to a manufacturer in accordance with examples;

[0006] FIG. 4 is a block diagram of a method of product verification in relation to a customer site in accordance with examples; and

[0007] FIG. 5 is a block diagram showing a tangible, non-transitory, computer-readable medium that stores code configured to direct a processor to perform product verification in accordance with examples.

**DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS**

**[0008]** Examples of the present techniques are directed to generating software for a computing device, and to product verification of the software and the computing device. The software may be generally including a software module(s) and may generally computer readable instructions. This discussion may use “software,” “software module,” and “software modules” interchangeably, and with the understanding that such will include computer readable instructions.

**[0009]** As for product verification of the software module, or of the computing device having the software module, a first indicator may be calculated as a function of the software module as generated and prior to the software module being installed on the computing device. A second indicator may be calculated as a function of the software module as installed on the computing device (e.g., with the software module as stored on the computing device at a manufacturer site). The second indicator may be compared to the first indicator. If the indicators are equal, then the computing may be fit to ship to the customer. On the other hand, if the second indicator is not equal to the first indicator, such may be an indication that the software module was compromised during the fabrication of the computing device at the manufacturer.

**[0010]** The computing device having the software module (e.g., if not compromised) may be shipped from the manufacturer to the customer. A third indicator may be calculated as a function of the software module as stored on the computing device as received by the customer. The third indicator may be compared to the first indicator. If the third indicator is not equal to the first indicator, such may be an indication that the computing device and/or software module was compromised while the computing device was in transit from the manufacturer to the customer, for example. Conversely, if the third indicator is equal to the first indicator, such may be an indication that the computing device (and software module) is fit to use by the customer.

**[0011]** The indicators may be hash values calculated via a secure hash function and based on product information of the software module. The product

information may include product type, product number, software module revision indicator, customer name, customer order number, and the like.

**[0012]** In sum, computing devices having firmware and/or embedded software modules may be compromised during manufacture or while in route to the customer. In response, the present techniques provide for a hash generation and verification process for firmware and embedded software modules on computing devices. A first hash value based on the firmware/software module is calculated during development of the firmware/software. A second hash value based on the firmware/software module as stored on the computing device is calculated during the manufacture of the computing device. The computing device is shipped to the customer if the second hash value is equal to the first hash value. A third hash value is calculated based on the firmware/software stored on the computing device as received by the customer. The customer may be notified that the computing device is acceptable to use if the third hash value is equal to the first hash value.

**[0013]** A particular example includes a Product Firmware Recipe Authentication Process (PF-RAP) which may be a hash generation and verification process for firmware and embedded software on products (software, computing devices, etc.). During the manufacturing process, a hash may be generated based on product type, product number, firmware revision, embedded software revision, and so on. This hash may be compared to a hash based on development originated data for the same configuration. The product may not be allowed to ship from the manufacturer to the customer unless the hash files are a match. Once the product is at the customer site, a third hash may be generated and compared to the development originated data. If there is a mismatch, the product may not be allowed to be used by the customer until the mismatch is resolved.

**[0014]** During the manufacturing process, a hash may be generated based on product type, product number, firmware revision, and embedded software revision. A Secure Hash Algorithm (SHA) function may be used to generate a hash file. This hash file may be sent to the secured PF-RAP web server, where

this hash file may be compared to a hash generated by same SHA function by development for the same configuration. If the hash is matched, a message may be sent to manufacturing authorizing the shipment of this product to the customer. Again, generally, a product may not be allowed to ship from the manufacturer to the customer unless hash files are a match. As mentioned, once the product is at the customer site, a third hash may be generated when the system is first powered up, for example. This hash may be sent to same secured PF-RAP web server, where this hash may be compared to the hash generated by the same SHA function by development organization for the same configuration. If the hash is matched, a message may be sent to the product system at the customer authorizing the use of product. However, as indicated, a product may not be allowed to be used by customer until any mismatch is resolved.

**[0015]** Advantages may include to protect the supplier or generator of the software from delivering compromised products to customer. Implementation may demonstrate to customers that the software developer and supplier is serious about protecting their customers from cyber risks such as malware, rootkit, and the like. In some examples, market share and revenue could be increased through reduced cyber risks product. In certain embodiments, the technique can be employed for products containing firmware and/or embedded software, such products as storage, servers, network, and so forth. The technique may facilitate use of the technique by both internal and external manufacturing organizations.

**[0016]** Continuing with the particular example, some of the various interacting entities are discussed. Such entities may include: (1) software development organization; (2) the PF-RAP center; (3) the manufacturer; and (4) the customer, etc.

**[0017]** The initial product development of the software by the development organization may include a validation of the product in creating individual product's firmware and embedded software's hash file using a PF-RAP client application. Product information and the hash file may be sent from development to the PF-RAP server. As indicated, the product information may

include product type, product number, firmware revision, firmware hash file, embedded software revision, embedded software file has file, and other information. The firmware and/or embedded software may be released from development to production (the fabricator or manufacturer).

**[0018]** The PF-RAP center may collect product firmware and embedded software hash file from development. The software hash file (including for individual product firmware, embedded software, etc.) may be stored as a hash file with read only attribute, for example. An individual product's hash file (e.g., to use in comparison for production process) may be available to a secure web site with read only attribute, for instance. The PF-RAP center may collect a product's firmware, embedded software, and system information bundle of hash file from production process (if matches). The PF-RAP center may store a product's firmware, embedded software, system's information bundle of hash file. This product's bundle hash file may be available (e.g., to use in comparison for customer) to a secure web site with read only attribute.

**[0019]** With respect to the manufacturer or production, the product or computing device having the software product may be built and tested, and including installing the software (e.g., firmware and embedded software). During production, a product's firmware and embedded software signature file may be created using an PF-RAP client. A comparison may be made between the manufacturer firmware and embedded software against PF-RAP database signature files using PF-RAP client. If the hash values do not match, notification may be provided to the development organization and/or the manufacturer, and production support staff waits for corrective action. If the hash values do match, a bundle of hash file (including firmwares, embedded softwares, and system information) may be created by the manufacturer using a PF-RAP client, for later comparison to customer information. The product may be sent to the customer.

**[0020]** As for the customer experience, the customer may receive a product from the manufacturer or factory. In particular examples, the customer may power-on the system and automatically trigger PF-RAP client as part of initial system boot process. A run-time bundle of hash file (including firmwares,

embedded softwares, and system information) may be created using an PF-RAP client. The product's hash value may be compared with factory originated hash value via secure web site using PF-RAP client, for instance. If the hash values match, the system may continue booting the remainder of the boot process with default operating system, and with completing operating system boot such as Windows, Linux, etc. If the hash values do not match, a message may be displayed, such as "please contact customer service," and the system does not proceed to the remainder of the boot process.

**[0021]** FIG. 1 is an example system 100 having a computing system (server 102) for product verification in relation to at least a manufacturer. The server 102 has product verification code 104 (e.g., instructions, logic, etc.) stored in memory 106 and executable by a processor 108. Further, the memory 106 or other memory may store one or more indicators 110 (e.g., hash values) calculated via the product verification code 104, as discussed below.

**[0022]** The memory 106 may include nonvolatile memory and volatile memory. The processor 108 may be more than one processor 108, and each processor 108 may have more than one core. The processor 108 as a hardware processor may be a central processing unit (CPU), a microprocessor, a controller, and so forth.

**[0023]** The product verification code 104 when executed by the processor 108 may provide for the server 102 to verify a product such as software 112 and/or a computing device having the software 112. As alluded to above, the software 112 is software module including computer readable instructions.

**[0024]** The illustrated embodiment depicts software 112 which may be generated by a supplier or development organization and stored in a memory 114 device, for example. The software 112 may be supplied to a fabricator or manufacturer 116 which installs the software 112 on a computing device 118. The software 112 and/or the computing device 118 may be specific to a particular customer in some examples.

**[0025]** The product verification code 104 may be executed by the processor 108 to: (1) calculate a first indicator as a function of product information of the software 112 prior to the software 112 being stored on the computing device

118; (2) calculate a second indicator 110 as a function of the product information of the software 112 with the software 112 as stored on the computing device 118; and (3) compare the first indicator to the second indicator 110 for product verification such as product verification of the computing device 118 and/or software 112 at the manufacturer 116 site (e.g., when the manufacturer is ready to ship the computing device 118 to a customer).

**[0026]** In examples, the second indicator equaling the first indicator may indicate that the software 112 was not compromised at the manufacturer 116 site. If so, the manufacturer 116 may ship the computing device 118 having the software 112 to the customer. On the other hand, the second indicator not equaling the first indicator may generally indicate that the software 112 has been compromised at the manufacturer 116 site. Therefore, if compromised, the manufacturer 116 may not ship the computing device 118 to the customer in examples.

**[0027]** The software 112 may be firmware or embedded software, or both, on the computing device 118. The product information used to calculate the first and second indicators (and subsequent indicators) may be product type, product number, software firmware revision (e.g., a revision number or version), and other information. The software revision may be a firmware revision, an embedded software revision, and the like. The product type and product number may refer to the software 112 and/or the computing device 118 as products, and may be specific to a particular customer.

**[0028]** In examples, the product information for calculating the first indicator may be provided to the server 102 by the supplier (developer) or generator of the software 112, e.g., with the software 112 as stored in the memory 114 device and prior to download or transfer of the software 112 to the manufacturer 116 to be installed on a computing device 118. The product information for calculating the second indicator may be provided to the server 102 from the manufacturer 116, such as when fabrication of the computing device 118 is complete and being tested, and is ready to be shipped to the customer. The manufacturer 112 may be given a product-verification client application to

facilitate communication with (and provision of product information to) the server 102.

**[0029]** Moreover, the product verification code 104 on the server 102 may include a secure hash function to calculate the indicators 110. Therefore, the first indicator may be a first hash value, and the second indicator may be a second hash value. The first and second indicators 110 (e.g., hash values) may be stored by the server 102 in the memory 106 (or other memory).

Furthermore, as indicated, the second indicator may be calculated based on the product information with the software 112 as stored on the computing device 118 at the manufacturer 116 site and prior to shipment of the computing device 118 to a customer.

**[0030]** FIG. 2 is an exemplary system 200 having the server 102 for product verification in relation to a customer 202 site. In examples, the server 102 may be used for product verification both in relation to a manufacturer 116 (FIG. 1) and relation to a customer 202. The code 104 may be executable by the processor 108 for the server 102 to: (1) calculate a third indicator as a function of the product information of the software 112 with the software as stored on the computing device 118 as received by the customer 202; (2) retain the third indicator (e.g., a hash value) in the memory 106; and (3) compare the third indicator to the first indicator. This comparison may be for product verification of the computing device 118 and software 112 as received by the customer 202 prior to the customer 202 placing the computing device 118 into full operation.

**[0031]** The customer 202 may provide to the server 102 the product information of the computing device 118 and software 112, for the computing device 118 as received by the customer 202. The customer 202 may be given a product-verification client application to facilitate communication with (and provision of product information to) the server 102. Indeed, the client application may be utilized for the computing device 118 at the customer 202 site to communicate product information to the server 102 when the computing device 118 being initially booted at the customer 202 site.

**[0032]** If the third indicator does not equal the first indicator in the comparison by the server 102, such may be an indication the computing device

118 and/or software 112 was compromised while in transit to the customer 112, for example. If so compromised, the computing device 118 may halt the boot, for example, and display a notification not to use the computing device 118. Other forms of notification are applicable. If the third indicator equals the first indicator, the computing device 118 may complete the initial boot as normal.

**[0033]** Lastly, with respect to FIGS. 1 and 2, as indicated, the first, second, and third indicators, may be numerical values, alphanumeric, and so on. Indeed, the code 104 may include a function that converts the aforementioned product information to the indicators (e.g., numeric or alphanumeric values). The product information input to the function may be numbers and/or text, etc. The function may provide for a mathematical transformation or a set of mathematical transformations. Moreover, the function may be a hash function. Indeed, the code 104 may include (or have access to) a cryptographic hash function (hashing algorithm). Therefore, the indicators may be hash values (output of the hash function). The values returned by a hash function may be called hash values, hash codes, hash sums, or simply hashes, and the like. Other terms for the output may include digest or tag, and so on. The product information as input to the hash function may be numbers and/or text, etc.

**[0034]** FIG. 3 is an exemplary method 300 of product verification in relation to a manufacturer. At block 302, software (software module) is created or generated, such as by a developer or supplier. The software may be intended as firmware and/or embedded software for a computing device, and may be intended for a particular customer. The software and/or the computing device may be specific to a particular customer in some examples. The software may be downloaded or otherwise transferred to a fabricator or manufacturer for installation of the software on the computing device such as in the fabrication or assembly of the computing device, and ultimately for the computing device having the software to be shipped to a customer.

**[0035]** At block 304, a first indicator (e.g., numerical value, alphanumeric, hash value, etc.) as a function of the software as generated is calculated, via a processor, prior to the software being stored on the computing device. At block

306, the software is provided to a manufacturer for installation of the software on a computing device, and ultimately for shipment of the computing having the software to a customer. The software may include firmware or embedded software, or both.

**[0036]** At block 308, a second indicator as a function of the software as stored on the computing device is calculated via the processor. The first and second indicators may be calculated based on product information of the software. The software product information may include product type, product number, firmware revision version, embedded software revision version, customer identification, customer order number or identifier, and so on. The second indicator may be calculated in response to the manufacturer testing and/or when the manufacturer is ready to ship the computing device to a customer. The product information for calculating the second indicator may be provided from the manufacturer, such as via a client application.

**[0037]** At block 310, the second indicator is compared, via the processor, to the first indicator. If the first and second indicators are equal, such may be an indication that the software was not compromised during the manufacture of the computing device. In response in examples of this situation with no compromise, the manufacturer will ship the computing device having the software to the customer. Conversely, if the first and second indicators are not equal, such may indicate the software was compromised during the manufacture of the computing device. In response in examples of this compromise, the manufacturer will not ship the computing device having the software to the customer. Indeed, the manufacturer may be notified not to ship the computing device to the customer.

**[0038]** In certain examples, the first indicator may be a first hash value, and the second indicator may be a second hash value. Indeed, the first and second indicators may be calculated using a hash function.

**[0039]** Moreover, again, the second indicator may be calculated as a function of the software as stored on the computing device at a manufacturer site and prior to shipment of the computing device to a customer. In fact, the first indicator and second indicator may be calculated as a function of product

information of the software, and wherein the second indicator is calculated as a function of product information (e.g., a software revision version, customer identification, etc.) of the software as stored on the computing device at a manufacturing site and prior to shipment to a customer.

**[0040]** FIG. 4 is an exemplary method 400 of product verification in relation to a customer site. In some examples, the method 400 may operate in conjunction with the method 300 of FIG. 3. The method may include shipping the computing device (having the software or software module) to the customer in response to the second indicator equaling the first indicator. At block 402, a third indicator is calculated, via the processor, as a function of the software (software module) as stored on the computing device as received by the customer. The third indicator may be a hash value calculated with a same hash function used to calculate the first and second indicators.

**[0041]** At block 404, the third indicator is compared, via the processor, to the first indicator. The comparison may occur during customer boot of the computing device. If the third indicator is equal to the first indicator, the customer may proceed with use of the computing device (and software), as indicated in block 406. In contrast, if the third indicator does not equal the first indicator, the customer may be notified to not use the computing device, as indicated in block 408.

**[0042]** With respect to FIGS. 3 and 4, the indicators (first, second, third) calculated may be numerical values or alphanumeric. A function may be employed to convert the product information to the indicators. Moreover, the indicators as calculated may be hash values (digest, hash function output, etc.). If so, the first indicator may be a first hash value, the second indicator may be a second hash value, and the third indicator is a third hash value. The product information of the software module as input to the cryptographic hash function may be text or numbers, and so on.

**[0043]** FIG. 5 is a block diagram showing a tangible, non-transitory, computer-readable medium that stores code configured to direct a processor to perform product verification. The computer-readable medium is referred to by the reference number 500. The computer-readable medium 500 can include

RAM, a hard disk drive, an array of hard disk drives, an optical drive, an array of optical drives, a non-volatile memory, a flash drive, a digital versatile disk (DVD), or a compact disk (CD), among others. The computer-readable medium 500 may be accessed by a processor 502 over a computer bus 504.

Furthermore, the computer-readable medium 500 may include code configured to perform the methods and techniques described herein. The various software components discussed herein may be stored on the computer-readable medium 500. A portion 506 of the computer-readable medium 700 can include product verification code, which may be executable code (machine readable instructions) that directs a processor or controller in performing product verification, including product verification of software, and/or of a computing device having the software (e.g., firmware, embedded software, etc.). The computer readable medium 500 may be the memory 106 in the server 102 of FIGS. 1 and 2, for example. The computer readable medium 500 may include the code 104 of FIGS. 1 and 2, for instance.

**[0044]** An example includes a tangible, non-transitory, computer-readable medium having instructions for product verification, the instructions direct a processor to: (1) calculate a first indicator as a function of product information of software (software module) prior to the software being stored on a computing device; (2) calculate a second indicator as a function of the product information of the software with the software as stored on the computing device; and (3) compare the first indicator to the second indicator. The software may be firmware or embedded software, or both. Moreover, the code in portion 506 may include a hash function (e.g., cryptographic hash function) to calculate the first and second indicators. Therefore, the indicators as calculated may be hash values (digest, hash function output, etc.). If so, the first indicator may be a first hash value, and the second indicator may be a second hash value. The product information of the software module as input to the cryptographic hash function may be text or numbers, and so on.

**[0045]** The second indicator is calculated as a function of the product information with the software as stored on the computing device at a manufacturer site and prior to shipment of the computing device to a customer.

Further, the instructions may direct the processor to: (1) calculate a third indicator as a function of the product information of the software with the software as stored on the computing device as received by the customer; and (2) compare the third indicator to the first indicator. The third indicator may also be a hash value as calculated via the hash function.

**[0046]** While the present techniques may be susceptible to various modifications and alternative forms, the exemplary examples discussed above have been shown only by way of example. It is to be understood that the technique is not intended to be limited to the particular examples disclosed herein. Indeed, the present techniques include all alternatives, modifications, and equivalents falling within the true spirit and scope of the appended claims.

**CLAIMS**

What is claimed is:

1. A method of product verification, comprising:  
generating software module for a computing device;  
calculating, via a processor, a first indicator as a function of the software module as generated and prior to the software module being stored on the computing device;  
calculating, via the processor, a second indicator as a function of the software module as stored on the computing device; and  
comparing, via the processor, the first indicator to the second indicator.
2. The method of claim 1, wherein the software module comprises computer readable instructions associated with firmware or embedded software, or both, wherein the first indicator is a first hash value, and wherein the second indicator is a second hash value.
3. The method of claim 1, wherein the first indicator comprises a first indicator or is a first alphanumerical indication, wherein the second indicator comprises a second indicator or is a second alphanumerical indication, and wherein the second indicator is calculated as a function of the software module as stored on the computing device at a manufacturer site and prior to shipment of the computing device to a customer.
4. The method of claim 3, comprising:  
shipping the computing device to the customer in response to the second indicator equaling the first indicator;  
calculating, via the processor, a third indicator as a function of the software module as stored on the computing device as received by the customer; and  
comparing, via the processor, the third indicator to the first indicator.

5. The method of claim 1, wherein the first indicator and second indicator are calculated as a function of product information of the software module, and wherein the second indicator is calculated as a function of the product information of the software module as store on the computing device at a manufacturing site and prior to shipment to a customer.

6. The method of claim 5, wherein the product information comprises a revision version of the software module.

7. A computer system for product verification, comprising:  
a processor; and  
memory comprising computer readable instructions for product verification of a computing device having a software module, the computer readable instructions executable by the processor for the computer system to:  
calculate a first indicator as a function of product information of the software module prior to the software module being stored on the computing device;  
calculate a second indicator as a function of the product information of the software module with the software module as stored on the computing device; and  
compare the second indicator to the first indicator.

8. The computer system of claim 7, wherein the computer system is a server, wherein the software module comprises computer readable instructions associated with a firmware or embedded software module of the computing device, or both, wherein the first indicator comprises a first hash value, and wherein the second indicator comprises a second hash value.

9. The computer system of claim 7, wherein the second indicator is calculated as a function of the product information with the software module as

stored on the computing device at a manufacturer site and prior to shipment of the computing device to a customer.

10. The computer system of claim 9, wherein the computer readable instructions are executable by the processor for the computer system to:

calculate a third indicator as a function of the product information of the software module with the software module as stored on the computing device as received by the customer; and  
compare the third indicator to the first indicator.

11. The computer system of claim 7, wherein the product information comprises a software module revision of the software module.

12. A tangible, non-transitory, computer-readable medium comprising computer readable instructions for product verification, the instructions direct a processor to:

calculate a first indicator as a function of product information of software module prior to the software module being stored on a computing device;  
calculate a second indicator as a function of the product information of the software module with the software module as stored on the computing device; and  
compare the second indicator to the first indicator.

13. The computer-readable medium of claim 12, wherein the software module comprises computer readable instructions associated with firmware or embedded software of the computing device, or both, wherein the first indicator comprises a first hash value, and wherein the second indicator comprises a second hash value.

14. The computer-readable medium of claim 12, wherein the second indicator is calculated as a function of the product information with the software

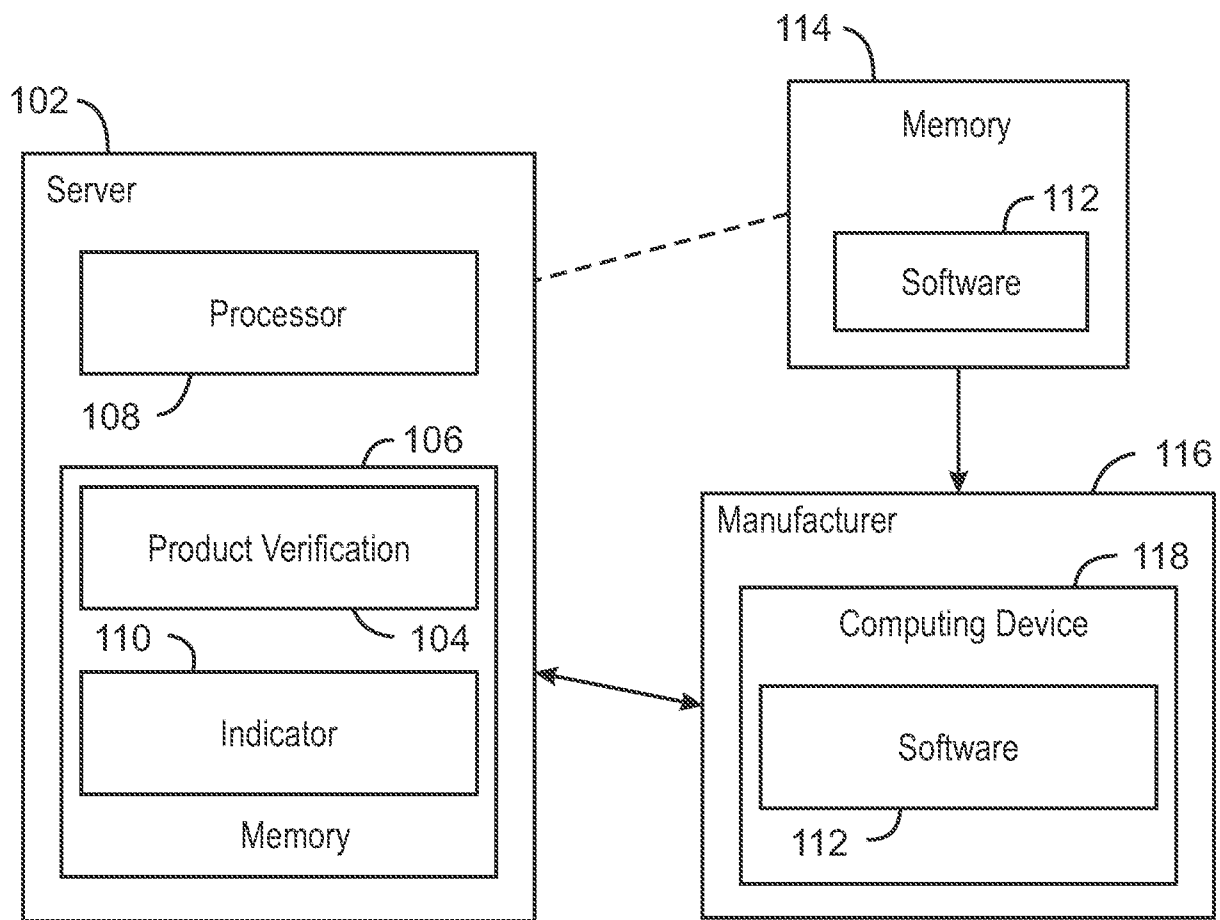
module as stored on the computing device at a manufacturer site and prior to shipment of the computing device to a customer.

15. The computer-readable medium of claim 14, wherein the instructions direct the processor to:

calculate a third indicator as a function of the product information of the software module with the software module as stored on the computing device as received by the customer; and

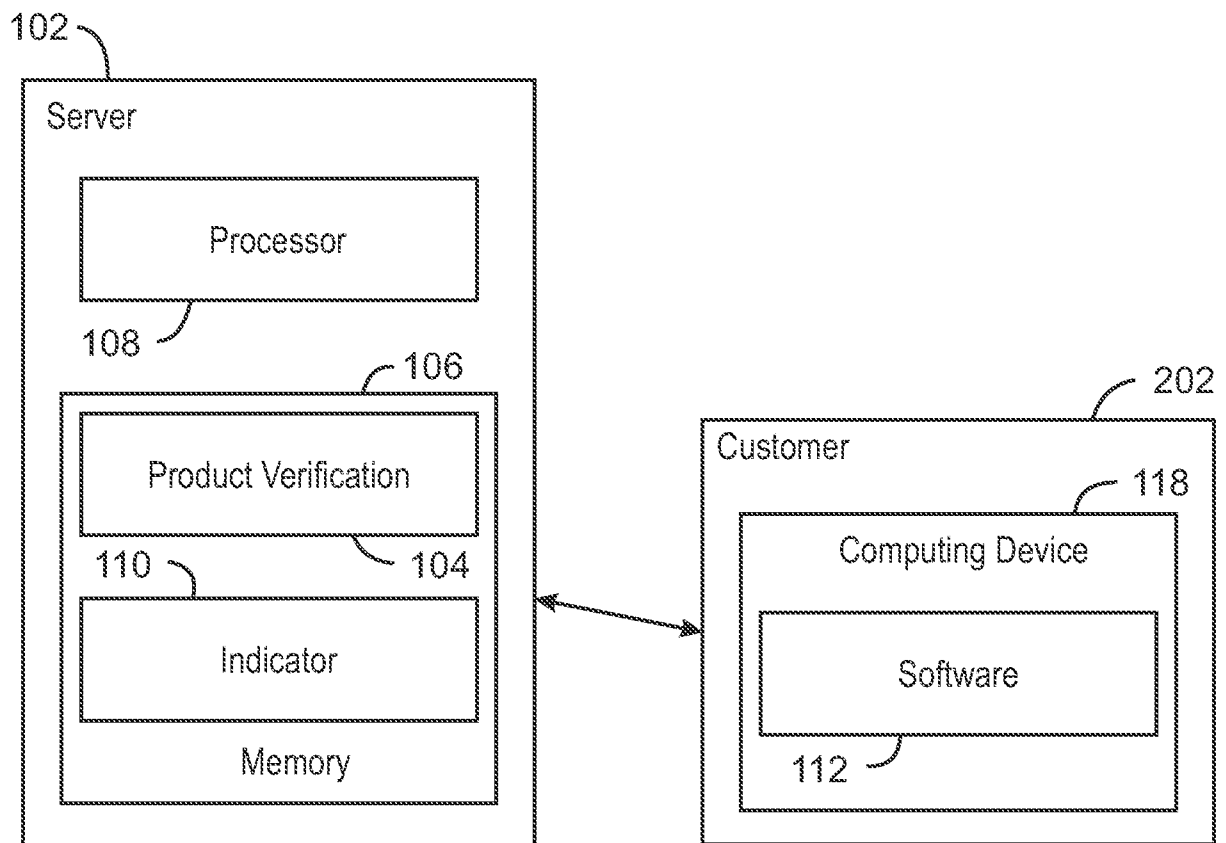
compare the third indicator to the first indicator.

1/5



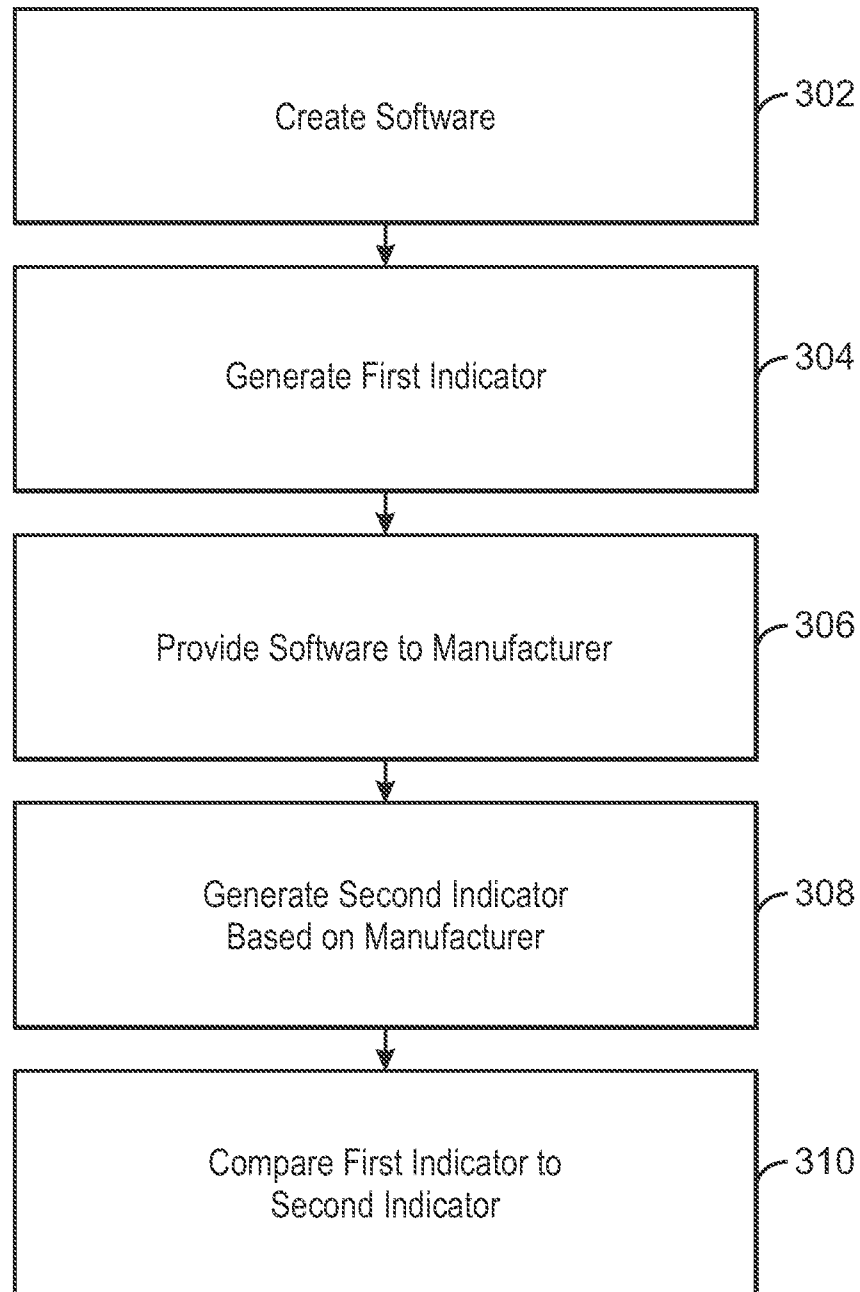
100  
FIG. 1

2/5



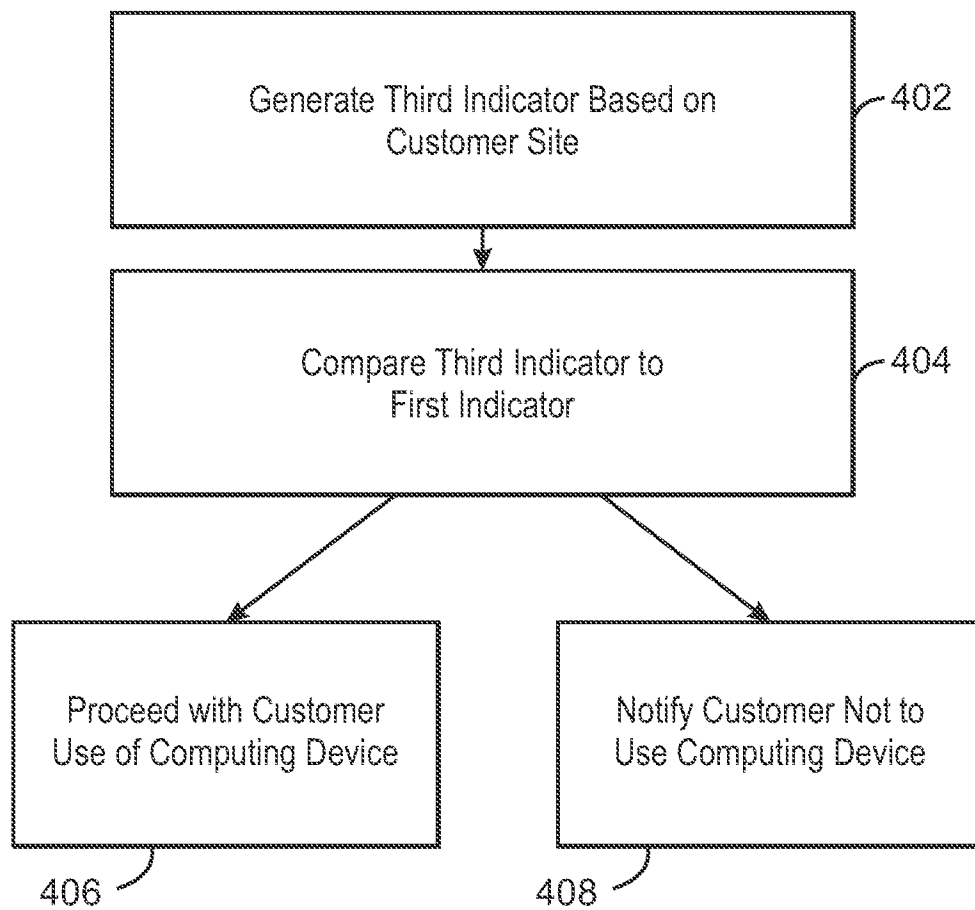
200  
FIG. 2

3/5



300  
FIG. 3

4/5



400  
FIG. 4



5/5

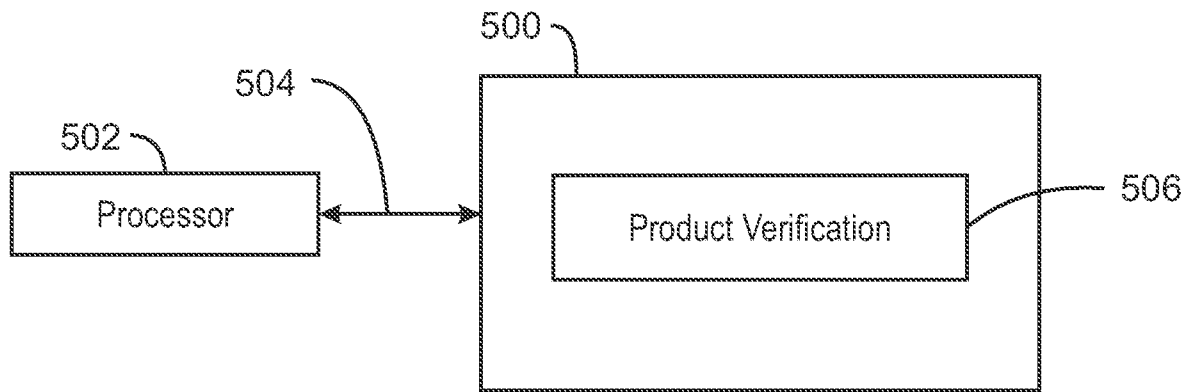


FIG. 5



**A. CLASSIFICATION OF SUBJECT MATTER****G06F 21/44(2013.01)i, G06F 21/12(2013.01)i, G06F 21/51(2013.01)i**

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)

G06F 21/44; G06F 17/00; G08B 21/00; H04L 9/06; H04B 1/74; G06F 21/60; G06F 11/36; G06F 21/12; G06F 21/51

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

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

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

eKOMPASS(KIPO internal) &amp; Keywords: product, verification, software, hash, indicator, calculate, compare, manufacture, customer, shipping

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category* | Citation of document, with indication, where appropriate, of the relevant passages                                                             | Relevant to claim No. |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------|
| Y         | US 2009-0187772 A1 (SEBASTIAN LANGE et al.) 23 July 2009<br>See paragraphs [0004]-[0006], [0014]-[0016], [0031], [0040], [0045]; and figure 2. | 1-15                  |
| Y         | US 2003-0126504 A1 (FINTAN RYAN et al.) 03 July 2003<br>See paragraphs [0011]-[0012], [0047]; and figure 4.                                    | 1-15                  |
| A         | EP 2557522 A2 (THE BOEING COMPANY) 13 February 2013<br>See paragraphs [0009], [0016]; and figure 1.                                            | 1-15                  |
| A         | US 2011-0227729 A1 (SATHIYAN PARAMESWARAN et al.) 22 September 2011<br>See paragraphs [0002], [0021]; and figure 9.                            | 1-15                  |
| A         | US 2012-0084608 A1 (MICHAEL PASTERNAK) 05 April 2012<br>See paragraphs [0011], [0030]; and figure 1.                                           | 1-15                  |



Further documents are listed in the continuation of Box C.



See patent family 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 application or patent 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

"&amp;" document member of the same patent family

Date of the actual completion of the international search

18 January 2017 (18.01.2017)

Date of mailing of the international search report

**19 January 2017 (19.01.2017)**

Name and mailing address of the ISA/KR

International Application Division

Korean Intellectual Property Office

189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea

Facsimile No. +82-42-481-8578

Authorized officer

CHIN, Sang Bum

Telephone No. +82-42-481-8398



**INTERNATIONAL SEARCH REPORT**

Information on patent family members

International application No.

**PCT/US2016/030136**

| Patent document<br>cited in search report | Publication<br>date | Patent family<br>member(s)                                                                 | Publication<br>date                                                |
|-------------------------------------------|---------------------|--------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| US 2009-0187772 A1                        | 23/07/2009          | US 2016-204945 A1<br>US 9262594 B2                                                         | 14/07/2016<br>16/02/2016                                           |
| US 2003-0126504 A1                        | 03/07/2003          | GB 2383854 A<br>GB 2383854 B<br>US 6990600 B2                                              | 09/07/2003<br>22/06/2005<br>24/01/2006                             |
| EP 2557522 A2                             | 13/02/2013          | CA 2781336 A1<br>CN 102915412 A<br>EP 2557522 A3<br>JP 2013-035539 A<br>US 2013-0036103 A1 | 04/02/2013<br>06/02/2013<br>20/02/2013<br>21/02/2013<br>07/02/2013 |
| US 2011-0227729 A1                        | 22/09/2011          | US 9177281 B2                                                                              | 03/11/2015                                                         |
| US 2012-0084608 A1                        | 05/04/2012          | US 9256488 B2                                                                              | 09/02/2016                                                         |