(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 3 642 751 B1

(12) EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention of the grant of the patent:
06.10.2021 Bulletin 2021/40

(21) Application number: **18731599.9**

(22) Date of filing: **21.05.2018**

(51) Int Cl.:
G06F 21/57 $^{(2013.01)}$ ***G06F 21/64*** $^{(2013.01)}$
H04L 9/32 $^{(2006.01)}$ ***H04L 29/06*** $^{(2006.01)}$

(86) International application number:
PCT/US2018/033581

(87) International publication number:
WO 2018/236508 (27.12.2018 Gazette 2018/52)

(54) MUTUAL AUTHENTICATION WITH INTEGRITY ATTESTATION

GEGENSEITIGE AUTHENTIFIZIERUNG MIT INTEGRITÄTSBESCHEINIGUNG

AUTHENTIFICATION MUTUELLE À ATTESTATION D’INTÉGRITÉ

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR

(30) Priority: **21.06.2017 US 201715629114**

(43) Date of publication of application:
29.04.2020 Bulletin 2020/18

(73) Proprietor: **Microsoft Technology Licensing, LLC**
Redmond, WA 98052-6399 (US)

(72) Inventors:
- **THOM, Stefan**
  Redmond, Washington 98052-6399 (US)
- **STEIN, Torsten**
  Redmond, Washington 98052-6399 (US)

(74) Representative: **CMS Cameron McKenna Nabarro Olswang LLP**
Cannon Place
78 Cannon Street
London EC4N 6AF (GB)

(56) References cited:
US-A1- 2004 127 196 **US-A1- 2011 179 268**
US-A1- 2016 054 989

Description

BACKGROUND

[0001] Consumer devices are increasingly configured with sensors, electronics, and networking capabilities to provide enhanced user experiences. Such devices utilize different software/firmware, applications, etc. to provide these enhanced experiences. Because the devices are connected to the network (e.g., via Wi-Fi, Ethernet, cellular network), cloud services push software/firmware updates to the device. Bad actors (e.g., hackers/malware) that have gained access to such devices are able to intercept such updates and falsely confirm updates to the device manufacturers or software providers. US 2011/179268 A1 describes one or more files of an application which are obtained and configured as a virtual storage volume. An application package is generated by encrypting, using a key, the one or more files configured as a virtual storage volume. A license generation module generates a license including both a usage policy for the application and the key. A computing device, to run the application, obtains and attempts to authenticate the application package. If the application package is authenticated, then a license associated with the application package is obtained and at least part of the application package is decrypted using the key in the license. A virtual storage volume that includes the application is mounted, and the application is executed in accordance with the usage policy in the license. However, if the application is not authenticated, then the application is not executed.
US 2016/054989 A1 describes an electronic device (such as a cellular telephone) which automatically installs and optionally personalizes a purposed application (which is sometimes referred to as an 'applet') on a secure element in the electronic device (which is sometimes referred to as 'applet creation'). In particular, when a digitally signed installation package containing the applet is received from an installing device (such as a server), the secure element verifies the digital signature of the installation package using an encryption key associated with a vendor of the secure element. Then, the secure element installs the applet. In addition, the secure element may optionally export user data from another applet installed on the secure element. Moreover, the secure element may personalize the installed applet using the user data from the other applet.
US 2004/127196 A1 describes a system and method to securely create, distribute, install and execute selected features of software on wireless devices combines three different types of licenses, a validation license, a digital rights management (DRM) license, and a feature license with a software application. Each of these three licenses work independent of each other, where the validation license helps prevent malicious code from executing on wireless devices, the DRM license prevents unauthorized copying of the software application and the feature license securely enables or disables specific features of the software application. The system also allows a wireless device to unwrap a DRM protected software application, to validate the software application, to enforce DRM usage rules and to execute selected features of the software application.

SUMMARY

[0002] According to aspects of the present invention there is provided a device, method, and storage media as defined in the accompanying claims.

[0003] In at least one implementation, a device includes a payload interface configured to receive a payload containing a sealed container, one or more provisioning code segments, and one or more policies based on the one or more provisioning code segments and corresponding to the sealed container. The device further includes a trusted computing manager configured generate one or more measurements of the one or more provisioning code segments according to the one or more policies, determine whether the generated one or more measurements satisfy the one or more policies, and unseal the sealed container responsive to the generated one or more measurements satisfying the one or more policies received in the payload. Satisfaction of the one or more policies confirms integrity of the device.

[0004] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

[0005] Other implementations are also described and recited herein.

BRIEF DESCRIPTIONS OF THE DRAWINGS

[0006]

FIG. 1 illustrates an example block diagram of device authentication using policies based on provisioning code.

FIG. 2 illustrates another example block diagram of device authentication using policies based on provisioning code.

FIG. 3 illustrates another example block diagram of device authentication using policies based on provisioning code.

FIG. 4 illustrates another example block diagram of device authentication using policies based on provisioning code.

FIG. 5 illustrates example operations for device authentication using policies based on provisioning code.

FIG. 6 illustrates example operations for device authentication using policies based on provisioning code.

FIG. 7 illustrates example operations for device authentication using policies based on provisioning code.

FIG. 8 illustrates example operations for device authentication using policies based on provisioning code.

FIG. 9 illustrates an example device that may be useful in implementing the described technology.

DETAILED DESCRIPTIONS

[0007] Consumer devices are increasingly configured with sensors, electronics, and networking capabilities to provide enhanced user experiences. Such devices utilize different software/firmware, applications, etc. to provide these enhanced experiences. Because the devices are connected to the network (e.g., via Wi-Fi, Ethernet, cellular network), cloud services push software/firmware updates to the device. The updates are provided to patch a security hole, fix a bug, or to enhance functionality. Generally, the updates are signed with a manufacturer or software provider key such that the device can that the update is received from an approved party. However, a bad actor (e.g., hacker/malware) that has control of the device may be able to falsely sign bad piece of software with the manufacturer/software provider key. Furthermore, the bad actor may send a deceptive confirmation back to the manufacturer/software provider that the pushed update was successfully installed. Furthermore, bad actors are able to detect differences between old software and new software to find the updates and exploit a new security patch, for example. Bad actors are further able to gain access to secure portions of these devices (e.g., trusted execution environment) and utilizes private keys to decrypt received encrypted payloads. These bad actors may further access private user data, user configuration data (e.g., Wi-Fi credentials), etc.

[0008] Implementations described herein include a smart device, connected device, IoT device, etc. configured with a trusted platform module (TPM) executing in a trusted execution environment (TEE). Software/firmware updates, user data, applications, etc. are pushed to the device as a payload. The payloads contain a sealed container (e.g., the software/firmware update, user data, applications, etc.), one or more policies, and one or more provisioning code segments corresponding to the one or more policies. The one or more provisioning code segments may be one or more agent code segments, which are code segments that implement some functionality or utilize some sensor implemented in the device. The one or more provisioning code segment may also be the code segments included in the sealed payload. The one or more policies include commands and/or conditions that must be satisfied before the device is able to unseal (e.g., release the sealed container to the device) the sealed container. The one or more policies are based at least on a measurement of the one or more provisioning code segments and may be based an output of execution of the one or more provisioning code segments. The policies are checked by the TPM of the device. If the measurement of the one or more provisioning code segments do not satisfy the policy, then the device is unable to unseal the sealed container and install the updates, utilize the data, etc. In other words, if the policies are not satisfied, the device may be compromised. If the policies are satisfied, the sealed container is unsealed and the device can install the updates, utilize the data, etc.

[0009] FIG. 1 illustrates an example block diagram 100 of device authentication using policies based on provisioning code. The block diagram 100 includes one or more provisioning services 102 (hereinafter "the provisioning service 102") that are configured to provision smart devices, such as the smart scale 124 Provisioning refers to providing customer or device specific configuration data (e.g., device setup information), software/firmware updates, customer data, customer applications, functionality data, etc. to the devices. The provisioning service 102 may be associated with a manufacturer or a device, a vendor, a stand-alone service, etc. The provisioning service 102 may have access to or manage user accounts and may manage or document devices using public keys associated with devices. The provisioning service 102 manages or documents ownership of such devices using the public key associated with the device.

[0010] In the illustrated implementation, the smart scale 124 includes firmware 114, and one or more agents 116, an operating system 122, and a trusted execution environment (TEE) 118. The agents 116 are code modules for operating one or more sensors, functionalities, etc. on the device. For example, a GPS sensor on the smart scale 124 includes agent code for operating the GPS sensor. Similarly, a temperature sensor includes agent code for operation of the temperature sensor. In another example implementation, one or more agent code segments are used to operate a Wi-Fi transceiver. Other agent segments are utilized for different functionality, such as, for example, code for weighing the user, displaying past user data at a user interface, etc. The device includes a trusted execution environment (TEE) 118 that communicates with a trusted platform module (TPM) 120. It should be understood that there may be more than one TPM 120.

[0011] The TPM 120 may be a microcontroller, such as a discrete silicon component in a semiconductor package, an integrated component incorporated in one or more semiconductor packages, or the TPM 120 may be a firmware based TPM running in a TEE on a general-purpose system on chip (SoC). In the illustrated implementation, the TPM 120 is a firmware based TPM executed in the TEE 118. However, it should be understood that the TPM 120 may be a microcontroller executed in a trusted computing manager 126. The instructions for TEE 118 may be stored in read only memory (ROM) or write once read many memory (WORM) and is separated from the rest of the programs (e.g., the operating system 122, firmware 114, agent 116) that are executing on a CPU of the device. Accordingly, secrets like private keys needed by the TPM 120 are not accessible by other programs unless the keys are authorized to be used by other programs or certain conditions are satisfied. The TPM 120 securely stores passwords, digital keys, and certificates, that provide unique identification and authentication. The TPM 120 may include or have access to non-volatile storage that stores keys and authorization data. The TPM 120 further includes one or more platform configuration registers (PCRs) that store measurements of the software 112 (including the agents 116, the firmware 114, and/or the operating system 122). The TPM 120 may further include one or more engines for encryption and hashing. For example, the TPM 120 includes a random number generator, a hashing engine (e.g., SHA-2), a key generator, a RSA engine, etc. The TPM 120 further includes an execution engine for executing different functionalities in the TPM 120.

[0012] When the smart scale 124 is booted, the boot code that is loaded (including the operating system and the firmware 114) are measured and recorded as integrity measurements in one or more of the PCRs of the TPM 120. The measurements are generated at the trusted computing manager 126 (and may be generated by the TEE 118). These measurements can be used as evidence for how a system started and to make sure that a TPM 120 based key is used only when the correct software (e.g., the software 112) was used to boot the system. Restricting utilization of a key based on the correct boot code starting the system is called "sealing" a key. The key seal is based on one or more policies. These policies direct the TPM 120 of what measurements to make or receive. If the measurements satisfy a policy (e.g., if a number of conditions are met), then the key is unsealed and may be used for an operation. Satisfaction of a policy is based on the measurements.

[0013] The one or more policies define a set of one or more controls or conditions that must be met before actions are authorized on a TPM 120 entity. The one or more controls or conditions can be combined using AND and OR combinations. An authorization policy digest is created by a combination of assertions (e.g., conditions). The authorization policy digest may be created outside the TPM 120 using software that emulates the policy calculations done on the TPM 120. In such implementations, the authorization policy digest may be created by the provisioning service 102 and sent with the payload 104. In other implementations, a trial policy can be created with the TPM. In a trial policy, the assertions (controls and/or conditions) pass and the authorization policy digest is created.

[0014] After the authorization policy digest is created, the one or more controls or conditions (e.g., policy commands) are sent to the TPM 120 and the TPM 120 hash-extends the commands to create a TPM policy digest. An authorization command (e.g., such as unseal the sealed container 110) is sent to the TPM 120. The TPM 120 determines whether the TPM policy digest matches the authorization policy digest. If so, the command is authorized and executed. In this example implementation, the sealed container 110 is unsealed by the authorization command. If the TPM policy digest does not match the authorization policy digest, then the authorization fails and the authorization command is not executed (e.g., the sealed container 110 is not unsealed).

[0015] It should be understood that the one or more policies 106 define commands to initiate a policy session, commands that are checked (e.g., measurements received), and the authorization command (e.g., unseal the sealed container 110). Furthermore, the one or more policies 106 may include the authorization policy digest if the authorization policy digest is created outside the TPM 120. Accordingly, the implementations described below that discuss determining whether the policies are "satisfied" include the operations described to create an authorization policy digest, process commands to create a TPM policy digest, and determine whether the digests match.

[0016] Furthermore, a command to "unseal" the sealed container 110 may be specified by authorizing an entity within the TPM 120, such as a key, to be utilized by an entity outside the TPM (e.g., the TEE 118 and/or the trusted computing manager 126). Thus, the implementations described that discuss "unsealing" the sealed container may include steps such as migrating a decryption key outside the TPM 120.

[0017] The provisioning information (e.g., data, software/firmware updates, applications) are send to the smart scale 124 as a payload 104. The payload 104 includes one or more policies 106, one or more provisioning code segments (provisioning code 108), and a sealed container 110. The sealed container 110 contains the provisioning information prepared for the device. The provisioning code 108 may code specific to one or more agents/functionality and thus may be considered one or more agent code segments. The sealed container 110 is released to the device if the one or more policies are satisfied. Thus, the payload 104 is received by a payload interface 128 of the device, and the provisioning code 108 is loaded in the agent(s) 116. The agent code 108 is executed by the smart scale 124 and generates an output which is communicated to the TPM 120 through the trusted computing manager 126. The TPM 120 receives a number of measurements, including at least a measurement of at least the received provisioning code 108. The measurements and the generated output of the provisioning code 108 are used to determine whether the received policies 106 are satisfied. If the policies are satisfied by the measurements of the provisioning code 108 and the output of execution of the provisioning code 108, then the sealed container 110 is unsealed and released to the smart scale 124. If the sealed container 110 is unsealed, then the smart scale 124 installs any software/firmware updates, any applications, or stores any data that is included in the sealed container 110. If the policies 106 are not satisfied, then the smart scale 124 may be compromised and the sealed container 110 is not unsealed. Because the one or more policies are satisfied by the measurements and/or the output, the smart scale 124 self-attests to the integrity of the device.

[0018] In some example implementations, the provisioning code 108 is the device specific update that is included in the sealed container 110. In such implementations, the TPM 120 may measure the provisioning code (e.g., software update) in the sealed container (e.g., conduct a hash integrity check, decrypt the provisioning code, etc.) based on the one or more policies 106. In such an implementation, if the policies are satisfied, the provisioning code 108 is released outside the TPM 120, the TEE 118, and/or the trusted computing manager 126 for execution one or more processors of the smart scale 124.

[0019] In some example implementations, the one or more polices 106 are further based on code already executable on the device (e.g., operating system/firmware not in the payload 104. In such an example implementation, the policies are based on projected measurements of the code executable on the device in addition to the provisioning code received in the payload 104.

[0020] At some point after the payload 104 is sent to the smart scale 124, the provisioning service 102 may initiate a device health attestation check. A measured boot sequence (that measures the boot code sequence stored on the PCRs of the TPM 120) may be securely sent to the provisioning service 102. The provisioning 102 checks the measured boot sequence to determine if the measured boot sequence is correct (e.g., the software is not compromised). If the measured boot sequence is correct, then it is confirmed that the device correctly unsealed the sealed container 110 and installed the data/code. If the measured boot sequence is not correct, then the provisioning service 102 may initiate corrective action with the device. Such corrective action may include, without limitation, instructing the device to wipe all software, bricking the device, or notifying the user that the device is compromised. If the user is notified of the device being compromised, the user may conduct an offline reset of the device or take the device in to a manufacturer or vendor for repairs and/or replacement.

[0021] The above described features allow the provisioning services 102 to push customized code/data to a number of devices and restrict utilization of the code/data based on satisfaction of the policies 106 sent with the payload 104. Furthermore, because the agent code 108 is sent with the payload 104, any infected device is not able to read the agent code to determine potential policy measurements before the agent code 108 is received. In other words, the agent code provides some functionality that the device is not "aware" of before the agent code 108 is received. FIGs. 2-4 illustrate different implementations of securing the payload 104.

[0022] FIG. 2 illustrates another example block diagram 200 of device authentication using policies based on provisioning code. The block diagram 200 includes one or more provisioning services 202 (hereinafter "the provisioning service 202") that are configured to provision smart devices, such as the device 206. Provisioning refers to providing tailored configuration data (e.g., device setup information), software/firmware updates, user data, user applications, functionality data, etc. to the devices. The provisioning service 202 may be associated with a manufacturer or a device, a vendor, a stand-alone service, etc. The provisioning service 202 may have access to or manage user accounts and may manage or document ownership of devices using public keys associated with devices.

[0023] The block diagram 200 further includes a device 206, which may be a smart device, internet of things (IoT) device, connected device, etc. In one example implementation, the device 206 is a smart scale. The device 206 includes software 220, which may include firmware, operating system code, agent code, etc. The device further includes a trusted platform module (TPM) 224 that executes in a trusted execution environment (TEE) 222. The TPM 224 securely stores passwords, digital keys, and certificates, that provide unique identification and authentication. The TPM 224 may include or have access to non-volatile storage that stores keys and authorization data. The TPM 224 further includes one or more platform configuration registers (PCRs) that store measurements of the software 220 (including agent code segments and firmware). The TPM 224 may further include one or more engines for encryption and hashing. For example, the TPM 224 includes a random number generator, a SHA-1 hashing engine, a key generator, a RSA engine, etc. The TPM 224 further includes an execution engine for executing different functionalities in the TPM 224.

[0024] The provisioning service 202 transmit a payload 204 to the device 206. The payload includes a sealed container 212 that includes data 214 and/or code 216 configured for the device 206. The data 214 may be user data, configuration data, etc., and the code 216 may include software/firmware updates, user applications, etc. The sealed container 212 is protected by one or more policies 210. The one or more policies 210 and the sealed container 212 are further part of an integrity protected container 208. They payload 204 further includes one or more agent code segments 218 that are outside the integrity protected container 208 of the payload 204.

[0025] The payload 204 is transmitted to the device 206 and the agent code is installed in the software 220. The integrity protected container 208 is processed by the trusted platform module 224 in the trusted execution environment 222. The integrity protected container 208 is protected by a hash algorithm, such as SHA-2. As such, the payload 204 includes a hash value that is encrypted using a public key that is associated with a private key stored in the trusted platform module. Thus, the trusted platform module decrypts the hash value, hashes the integrity protected container 208 (which includes the one or more policies 210 and the sealed container 212) to determine a second hash value. If the hash value and the second hash value match, then the integrity protected container 208 has integrity. In other words, the integrity protected container 208 has not been altered during transmission from the provisioning service 202 to the device 206.

[0026] The trusted platform module 224 then takes measurements of the one or more agent code segments 218. Furthermore, the one or more agent code segments 218 are executed, which generates an output. The trusted platform module 224 receives the output and measurements and stores them in the one or more platform configuration registers (PCRs) of the trusted platform module 224. The trusted platform module 224 determines whether the one or more policies 210 are satisfied by the values stored in the one or more PCRs of the trusted platform module 224. If the one or more policies 210 are satisfied, then the sealed container 212 is unsealed and released to the device 206. If the one or more policies 210 are not satisfied, the trusted platform module 224 is unable to unseal the sealed container 212. Accordingly, the data 214 is not stored and/or the code 216 is not installed.

[0027] FIG. 3 illustrates another example block diagram 300 of device authentication using policies based on provisioning code. The block diagram 300 includes one or more provisioning services 302 (hereinafter "the provisioning service 302") that are configured to provision smart devices, such as the device 306. Provisioning refers to providing tailored configuration data (e.g., device setup information), software/firmware updates, user data, user applications, functionality data, etc. to the devices. The provisioning service 302 may be associated with a manufacturer or a device, a vendor, a stand-alone service, etc. The provisioning service 302 may have access to or manage user accounts and may manage or document devices using public keys associated with devices.

[0028] The block diagram 300 further includes a device 306, which may be a smart device, internet of things (IoT) device, connected device, etc. In one example implementation, the device 306 is a smart scale. The device 306 includes software 320, which may include firmware, operating system code, agent code, etc. The device further includes a trusted platform module (TPM) 324 that executes in a trusted execution environment (TEE) 322. The TPM 324 securely stores passwords, digital keys, and certificates, that provide unique identification and authentication. The TPM 324 may include or have access to non-volatile storage that stores keys and authorization data. The TPM 324 further includes one or more platform configuration registers (PCRs) that store measurements of the software 320 (including agents and firmware). The TPM 324 may further include one or more engines for encryption and hashing. For example, the TPM 324 includes a random number generator, a SHA-1 hashing engine, a key generator, a RSA engine, etc. The TPM 324 further includes an execution engine for executing different functionalities in the TPM 324.

[0029] The provisioning service 302 transmit a payload 304 to the device 306. The payload 304 includes one or more policies 308 and an encrypted payload 310. In the payload 304, the one or more policies 308 are outside the encrypted payload 310. The encrypted payload 310 includes a sealed container 312 and one or more agent code segments 318. The sealed container 312 includes data 314 and/or code 316 configured for the device 306. The data 314 may be user data, configuration data, etc., and the code 316 may include software/firmware updates, user applications, etc. The sealed container 312 is protected by the one or more policies 308.

[0030] The payload 304 is transmitted to the device 306 and the one or more policies 308 are delivered to the TPM 324. In some implementations, the encrypted payload 310 is encrypted with a public key associated with a private key known by the TPM 324. In some other implementations, the encrypted payload 310 is encrypted with a symmetric key, which itself is encrypted by a public key associated with a private key known by the TPM 324. The TPM 324 utilizes the known private key to decrypt the encrypted payload 310 (or decrypt the symmetric key, which is then used to decrypt the encrypted payload 310). The one or more agent code segments 318 are then released to the software 320.

[0031] The trusted platform module 324 then takes measurements of the one or more agent code segments 318. Furthermore, the one or more agent code segments 318 are executed, which generates an output. The trusted platform module 324 receives the output and measurements and stores them in the one or more platform configuration registers (PCRs) of the trusted platform module 324. The trusted platform module 324 determines whether the one or more policies 308 are satisfied by the values stored in the one or more PCRs of the trusted platform module 324. If the one or more policies 308 are satisfied, then the sealed container 312 is unsealed and released to the device 306. If the one or more policies 308 are not satisfied, the trusted platform module 324 is unable to unseal the sealed container 312. Accordingly, the data 314 is not stored and/or the code 316 is not installed.

[0032] FIG. 4 illustrates another example block diagram of device authentication using policies based on provisioning code. The block diagram 400 includes one or more provisioning services 402 (hereinafter "the provisioning service 402") that are configured to provision smart devices, such as the device 406. Provisioning refers to providing tailored configuration data (e.g., device setup information), software/firmware updates, user data, user applications, functionality data, etc. to the devices. The provisioning service 402 may be associated with a manufacturer or a device, a vendor, a stand-alone service, etc. The provisioning service 402 may have access to or manage user accounts and may manage or document devices using public keys associated with devices.

[0033] The block diagram 400 further includes a device 406, which may be a smart device, internet of things (IoT) device, connected device, etc. In one example implementation, the device 406 is a smart scale. The device 406 includes software 420, which may include firmware, operating system code, agent code, etc. The device further includes a trusted platform module (TPM) 424 that executes in a trusted execution environment (TEE) 422. The TPM 424 securely stores passwords, digital keys, and certificates, that provide unique identification and authentication. The TPM 424 may include or have access to non-volatile storage that stores keys and authorization data. The TPM 424 further includes one or more platform configuration registers (PCRs) that store measurements of the software 420 (including agents and firmware). The TPM 424 may further include one or more engines for encryption and hashing. For example, the TPM 424 includes a random number generator, a SHA-1 hashing engine, a key generator, a RSA engine, etc. The TPM 424 further includes an execution engine for executing different functionalities in the TPM 424.

[0034] The provisioning service 402 transmit a payload 404 to the device 406. The payload 404 includes an encrypted payload 408 that includes one or more active policies 410 and a sealed container 412. The sealed container includes data 414 and/or code 416 configured for the device 406. The data 414 may be user data, configuration data, etc., and the code 416 may include software/firmware updates, user applications, etc. The TPM 424 decrypts the encrypted payload 408 using a private key stored in the TPM 424 (or an asymmetric key decrypted using the private key). The one or more active policies are executed by the device 406. If the one or more active policies are executed correctly then the TPM unseals the sealed container 412 and the data 414 and/or code are released to the device 406.

[0035] FIG. 5 illustrates example operations 500 for device authentication using policies based on provisioning code. A receiving operation 502 receives a payload containing a sealed container, one or more policies, and one or more agent code segments corresponding to the one or more policies. The one or more policies and the sealed container are integrity protected. Accordingly, the payload contains an encrypted hash value. The hash value is encrypted using a public key associated with a private key stored on the device. A decrypting operation 504 decrypts the encrypted hash value. A hashing operation 506 hashes the one or more policies and the sealed container to generate another hash value (a new hash value). A determining operation 508 determines whether the one or more policies and the sealed container have integrity. In other words, the determining operation 508 determines whether the one or more policies and the sealed container have been altered by comparing the decrypted hash value to the generated another hash value. If the one or more policies and the sealed container have been altered (e.g., the one or more policies and the sealed container do not have integrity), then the process continues to a waiting operation 518 that waits for additional instructions. Additional instructions may be received from a provisioning service to lock down or wipe the device.

[0036] If the one or more policies and the sealed container are not altered (e.g., the one or more policies and the sealed container have integrity/the hash values match), then an executing operation 510 executes the one or more agent code segments to generate an output. A measuring operation 512 measures the one or more agent code segments based on the one or more policies. The generated output and the measurement of the one or more agent code segments may be stored in one or more platform configuration registers (PCRs) of a trusted platform module (TPM) of the device. A determining operation 514 determines whether the one or more policies are satisfied by the measurement and the output. If the policies are not satisfied, then the sealed container is not unsealed and the waiting operation 518 waits for additional instructions. The device may be compromised and the device does not receive any data/code in the sealed container. If the policies are satisfied, the trusted platform module unseals (in an unsealing operation 516) the sealed container (e.g., using a key), and the data/code in the sealed container is released to the device. In other words, the device is not compromised.

[0037] FIG. 6 illustrates example operations 600 for device authentication using policies based on provisioning code. A receiving operation 602 receives a payload containing a sealed container, one or more policies, and one or more agent code segments corresponding to the one or more policies. The sealed container and the one or more agent code segments are encrypted. In some implementations, the sealed container and the one or more agent code segments are encrypted with a public key associated with a private key known by the trusted platform module (TPM) of the device. In some other implementations, the sealed container and the one or more agent code segments are encrypted with a symmetric key, which itself is encrypted by a public key associated with a private key known by the TPM. The TPM utilizes the known private key in a decrypting operation 604 that decrypts the encrypted sealed container and the one or more agent code segments (or decrypt the symmetric key, which is then used to decrypt the sealed container and the one or more agent code segments). An executing operation 606 executes the one or more agent code segments to generate an output. A measuring operation 608 measures the one or more agent code segments based on the one or more policies. A determining operation 610 determines whether the one or more policies are satisfied by the measurement and the output. If the policies are not satisfied, then the sealed container is not unsealed and a waiting operation 614 waits for additional instructions. The device may be compromised and the device does not receive any data/code in the sealed container. If the policies are satisfied, the trusted platform module unseals (in an unsealing operation 612) the sealed container (e.g., using a key), and the data/code in the sealed container is released to the device. In other words, the device is not compromised.

[0038] FIG. 7 illustrates example operations 700 for device authentication using policies based on provisioning code. Specifically, FIG. 7 illustrates the operations from the perspective of a provisioning service. A receiving operation 702 receives a communication from a device with a public device identification. A locating operation 704 locates a data, and/or software or firmware updates for the device using the public device ID. The public device ID may be documented in an ownership record that is connected to a customer account. The data and/or software or firmware updates may be specific to the device and or the customer with the device. Thus, the provisioning service utilizes the ownership record and the connected customer account/profile to generate the data and/or software or firmware updates. The data and/or software or firmware updates may be pre-prepared by the provisioning service and/or generated after the communication is received in the receiving operation 702. Furthermore, the data and/or software or firmware updates may be prepared for the device without receiving a communication in the receiving operation 702.

[0039] An identifying operation 706 identifies one or more agent code segments for policies. These agent code segments may be based on the type of device, functionality of the device, etc. A generating operation 708 generates one or more policies based on the one or more agent code segments. The generating operation 708 may include generating an authorization digest by simulating a TPM of the device. The generating operation 708 may further include generating commands for the policies, an authorization command, and/or combining (e.g., generating a hash chain) of the commands using AND and/or OR operations. A preparing operation 710 prepares a sealed container with the data and/or software or firmware updates using the one or more policies and the one or more agent code segments. The preparing operation 710 generating an authorization digest by simulating a TPM of the device. The preparing operation 710 may further include generating commands for the policies, an authorization command, and/or combining (e.g., generating a hash chain) of the commands using AND and/or OR operations.

[0040] A securing operation 712 secures the sealed container, the one or more policies, and the one or more agent code segments as a payload. The securing operation 712 may include encrypting (e.g., using the device's public key) one or more of the policies, agent code segments, and the sealed container. The securing operation 712 may further include hashing one of the payload portions to generate a hash value, and encrypting the hash value. A transmitting operation 714 transmits the secured payload to the device. The provision service may subsequently request a device attestation from the device. The device may send a measured boot sequence to the provisioning service. If the measured boot sequence does not satisfy an expected boot sequence, then the provisioning service may take corrective action with the device.

[0041] FIG. 8 illustrates example operations for device authentication using policies based on provisioning code. A receiving operation receives a payload containing a sealed container, one or more provisioning cod segments, and one or more policies. A generating operation 804 generates one or more measurements of the one or more provisioning code segments based on the one or more policies. The generating operation may be performed by a trusted computing manager and/or a trusted executing operation. A determining operation 806 determines whether the one or more policies are satisfied by the generated measurement. If the policies are satisfied by the generated measurement, an unsealing operation 808 unseals the sealed container to release the code and/or data to the device. If the policies are not satisfied, a waiting operation 810 waits for additional instructions. The additional instructions may include a device attestation check by an attestation service (e.g., determines whether the code/and or data is installed), or another device/service. The device or user may receive a notification indicating the device has been compromised and corrective action should be taken.

[0042] In some example implementations, the one or more provisioning code segments are the code segments included in the sealed container. In such an implementation, the policies may indicate certain operations to perform on the provisioning code segments (e.g., hash, decrypt, etc.). If the operations are performed by the device (e.g., a trusted platform module, trusted execution environment, and/or a trusted computing manager), then the policies are satisfied and the provisioning code is released to execute on the device.

[0043] FIG. 9 illustrates an example system (labeled as a processing system 900) that may be useful in implementing the described technology. The processing system 900 may be a client device, such as a laptop, mobile device, desktop, tablet, or a server/cloud device. The processing system 900 includes one or more processor(s) 902, and a memory 904. The memory 904 generally includes both volatile memory (e.g., RAM) and non-volatile memory (e.g., flash memory). An operating system 910 resides in the memory 904 and is executed by the processor 902. The memory 904 includes a read only memory (ROM) 914, which may be a write once, read many (WORM) memory.

[0044] One or more application programs 912 modules or segments, such as user applications 942, agent code 944, a trusted execution environment 946, and a trusted platform module 948, a trusted computing manager (not shown) are loaded in the memory 904 and/or storage 920 and executed by the processor 902. Firmware, one or more agents, a payload interface, and a policy manager (not shown) may also be loaded in the memory 904 and executed by the processor 902. The trusted execution environment 946 is stored in the ROM 914 (or

WORM) and executed by the processor 902. Data, such as user data, digests, measurements, keys, passwords, root secrets, etc. may be stored in the memory 904 or storage 920 and may be retrievable by the processor 902 for use in the by the user applications 942, the agent code 944, the trusted execution environment 946, the trusted platform module 948, etc. The storage 920 may be local to the processing system 900 or may be remote and communicatively connected to the processing system 900 and may include another server. The storage 920 may store resources that are requestable by client devices (not shown).

[0045] The processing system 900 includes a power supply 916, which is powered by one or more batteries or other power sources and which provides power to other components of the processing system 900. The power supply 916 may also be connected to an external power source that overrides or recharges the built-in batteries or other power sources.

[0046] The processing system 900 may include one or more communication transceivers 930 which may be connected to one or more antenna(s) 932 to provide network connectivity (e.g., mobile phone network, Wi-Fi®, Bluetooth®, etc.) to one or more other servers and/or client devices (e.g., mobile devices, desktop computers, or laptop computers). The processing system 900 may further include a network adapter 936, which is a type of communication device. The processing system 900 may use the network adapter 936 and any other types of communication devices for establishing connections over a wide-area network (WAN) or local-area network (LAN). It should be appreciated that the network connections shown are exemplary and that other communications devices and means for establishing a communications link between the processing system 900 and other devices may be used.

[0047] The processing system 900 may include one or more input devices 934 such that a user may enter commands and information (e.g., a keyboard or mouse). These and other input devices may be coupled to the server by one or more interfaces 938, such as a serial port interface, parallel port, universal serial bus (USB), etc. The processing system 900 may further include a display 922, such as a touch screen display.

[0048] The processing system 900 may include a variety of tangible processor-readable storage media and intangible processor-readable communication signals. Tangible processor-readable storage can be embodied by any available media that can be accessed by the processing system 900 and includes both volatile and nonvolatile storage media, removable and non-removable storage media. Tangible processor-readable storage media excludes intangible communications signals and includes volatile and nonvolatile, removable and non-removable storage media implemented in any method or technology for storage of information, such as processor-readable instructions, data structures, program modules or other data. Tangible processor-readable storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CDROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other tangible medium which can be used to store the desired information and which can be accessed by the processing system 900. In contrast to tangible processor-readable storage media, intangible processor-readable communication signals may embody computer-readable instructions, data structures, program modules or other data resident in a modulated data signal, such as a carrier wave or other signal transport mechanism. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, intangible communication signals include signals traveling through wired media, such as a wired network or direct-wired connection, and wireless media, such as acoustic, RF, infrared, and other wireless media.

[0049] Some embodiments may comprise an article of manufacture. An article of manufacture may comprise a tangible storage medium to store logic. Examples of a storage medium may include one or more types of processor-readable storage media capable of storing electronic data, including volatile memory or non-volatile memory, removable or non-removable memory, erasable or non-erasable memory, writeable or rewriteable memory, and so forth. Examples of the logic may include various software elements, such as software components, programs, applications, computer programs, application programs, system programs, machine programs, operating system software, middleware, firmware, software modules, routines, subroutines, operation segments, methods, procedures, software interfaces, application program interfaces (API), instruction sets, computing code, computer code, code segments, computer code segments, words, values, symbols, or any combination thereof. In one embodiment, for example, an article of manufacture may store executable computer program instructions that, when executed by a computer, cause the computer to perform methods and/or operations in accordance with the described embodiments. The executable computer program instructions may include any suitable type of code, such as source code, compiled code, interpreted code, executable code, static code, dynamic code, and the like. The executable computer program instructions may be implemented according to a predefined computer language, manner or syntax, for instructing a computer to perform a certain operation segment. The instructions may be implemented using any suitable high-level, low-level, object-oriented, visual, compiled and/or interpreted programming language.

[0050] The implementations described herein are implemented as logical steps in one or more computer systems. The logical operations may be implemented (1) as a sequence of processor-implemented steps executing in one or more computer systems and (2) as interconnected machine or circuit modules within one or more computer systems. The implementation is a matter of choice, dependent on the performance requirements of the computer system being utilized. Accordingly, the logical operations making up the implementations described herein are referred to variously as operations, steps, objects, or modules. Furthermore, it should be understood that logical operations may be performed in any order, unless explicitly claimed otherwise or a specific order is inherently necessitated by the claim language.

Claims

1. A device (124) comprising:

one or more processors (902);
a payload interface (128) executable by the one or more processors (902) to receive a payload (104) containing a sealed container (110), one or more provisioning code segments (108) outside the sealed container (110), and one or more policies (106) located outside the sealed container (110), the one or more policies (106) being based on the one or more provisioning code segments (108) and corresponding to the sealed container (110); and
a trusted computing manager (126) executable by the one or more processors (902) to:

generate (804) one or more measurements of the one or more provisioning code segments (108) according to the one or more policies (106), to determine (806) whether the generated one or more measurements satisfy the one or more policies (106) based on the one or more measurements and an output of execution of the one or more provisioning code segments (108), wherein the execution of the one or more provisioning code segments located outside the sealed container by the device (124) generates the output; and
unseal (808), at a trusted platform module (TPM) (120) of the trusted computing manager (126), the sealed container (110) responsive to determining that the generated one or more measurements and the output of the execution satisfy the one or more policies (106) received in the payload (104), satisfaction of the one or more policies (106) confirming integrity of the device (124).

2. The device (124) of claim 1 wherein the sealed container (110) and the one or more policies (106) are protected by a hash value and hash algorithm, the hash value being encrypted using a public key associated with a private key stored in the trusted platform module (TPM) (120) of the device (124), the trusted computing manager (126) further executable to decrypt the hash value using the private key stored on the TPM (120) and associated with the public key used to encrypt the hash value, hash the one or more policies (106) and the sealed container (110) to generate a new hash value; compare the hash value the new hash value to determine integrity of the sealed container (110) and the one or more policies (106).

3. The device (124) of claim 1 wherein the sealed container (110) and the one or more provisioning code segments are encrypted using a public key associated with a private key stored in the trusted platform module (TPM) (120), the trusted computing manager (126) further executable to decrypt the sealed container (110) and the one or more provisioning code segments using the private key stored in the TPM (120).

4. The device (124) of claim 1 wherein the one or more provisioning code segments (108) are active policies (410) executable by the device (124) to satisfy the one or more policies (106).

5. The device (124) of claim 1 wherein the trusted computing manager includes a firmware based trusted platform module (TPM) (120) executed in a trusted execution environment (TEE) (322) of the device (124).

6. The device (124) of claim 1 wherein the one or more policies (106) are further based on one or more projected measurements of code executable on the device (124), the code executable on device (124) stored on the device (124) prior to receiving the payload (104).

7. A method (880) of confirming integrity of a device (124) comprising:

receiving (802), at the device (124), a payload (104) containing a sealed container (110), one or more provisioning code segments (108) located outside the sealed container (110), and one or more policies (106) located outside the sealed container (110), the one or more policies (106) being based on the one or more provisioning code segments (108) and corresponding to the sealed container (110);
generating (804), at the device (124), one or more measurements of the one or more provisioning code segments (108) based on the one or more policies (106);
executing the one or more provisioning code segments (108) located outside the sealed con-

tainer (110) by the device (124) to generate an output; and
unsealing (808), at a trusted platform module (TPM) (120) of a trusted computing manager (126) of the device (124), the sealed container (110) for the device (124) responsive to determining (806) that the generated one or more measurements and the generated output satisfy the one or more policies (106), satisfaction of the one or more policies (106) confirming integrity of the device (124).

8. The method (800) of claim 7 wherein the sealed container (110) and the one or more policies (106) are protected by a hash value and hash algorithm, the hash value being encrypted using a public key associated with a private key stored in the trusted platform module (TPM) (120) of the device (124), method (800) further comprising:

decrypting (504) the hash value using the private key stored on the TPM (120) and associated with the public key used to encrypt the hash value;
hashing (506) the one or more policies (106) and the sealed container (110) to generate a new hash value; and
comparing (508) the hash value with the new hash value to determine integrity of the sealed container (110) and the one or more policies (106).

9. The method (800) of claim 7, wherein the sealed container (110) and the one or more provisioning code segments (108) are encrypted using a public key associated with a private key stored in the trusted platform module (TPM) (120) of the device (124), the method (800) further comprising:
decrypting (604) the sealed container (110) and the one or more provisioning code segments (108) using the private key stored in the TPM (120).

10. The method (800) of claim 8 wherein the one or more provisioning code segments (108) are active policies (410) executable by the device (124) to satisfy the one or more policies (106).

11. The method (800) of claim 7 wherein the device (124) includes a firmware based platform module (TPM) (120) executed in a trusted execution environment (322) of the device (124).

12. The method (800) of claim 7 wherein the one or more policies (106) are further based on one or more projected measurements of code executable on the device (124), the code executable on device (124) stored on the device prior to receiving the payload (104).

13. One or more tangible processor-readable storage media (904) embodied with instructions for executing on one or more processors (902) and circuits of a device (124) a process (800) comprising:

receiving (802) a payload (104) containing a sealed container (110), one or more provisioning code segments (108) located outside the sealed container (110), and one or more policies (106) located outside the sealed contained, the one or more policies (106) being based on the one or more provisioning code segments (108) and corresponding to the sealed container (110);
generating (804) one or more measurements of the one or more provisioning code segments (108) based on the one or more policies (106);
executing the one or more provisioning code segments (108) located outside the sealed container (110) by the device (124) to generate an output; and
unsealing (808), at a trusted platform module (TPM) (120) of a trusted computing manager (126) of the device (124), the sealed container (110) for the device (124) responsive to determining (806) by the TPM (120) that the generated one or more measurements satisfy the one or more policies (106) based on an output of execution by the one or more provisioning code segments (108), satisfaction of the one or more policies (106) confirming integrity of the device (124).

14. The one or more tangible processor-readable storage media (904) of claim 13 wherein the sealed container (110) and the one or more policies (1060) are protected by a hash value and hash algorithm, the hash value being encrypted using a public key associated with a private key stored in the trusted platform module (TPM) (120) of the device, the process further comprising:

decrypting (504) the hash value using the private key stored on the TPM (120) and associated with the public key used to encrypt the hash value;
hashing (506) the one or more policies (106) and the sealed container (110) to generate a new hash value; and
comparing (508) the hash value with the new hash value to determine integrity of the sealed container (110) and the one or more policies (106).

15. The one or more tangible processor-readable storage media (904) of claim 13. wherein the sealed container (110) and the one or more provisioning code segments (108) are encrypted using a public key associated with a private key stored in the trusted plat-

form module (TPM) (120) of the device (124), the process further comprising:
decrypting (604) the sealed container (110) and the one or more provisioning code segments (108) using the private key stored in the TPM (120).

Patentansprüche

1. Vorrichtung (124), umfassend:

   einen oder mehrere Prozessoren (902);
   eine Nutzlastschnittstelle (128), die durch den einen oder die mehreren Prozessoren (902) ausführbar ist, um eine Nutzlast (104) zu empfangen, die einen verplombten Container (110), ein oder mehrere Provisioning-Code-Segmente (108) außerhalb des verplombten Containers (110) und eine oder mehrere sich außerhalb des verplombten Containers (110) befindende Richtlinien (106) enthält, wobei die eine oder die mehreren Richtlinien (106) auf dem einen oder den mehreren Provisioning-Code-Segmenten (108) basieren und mit dem verplombten Container (110) übereinstimmen; und
   einen Trusted-Computing-Manager (126), der von dem einen oder den mehreren Prozessoren (902) ausführbar ist, um:
   eine oder mehrere Messungen des einen oder der mehreren Provisioning-Code-Segmente (108) gemäß der einen oder den mehreren Richtlinien (106) zu generieren (804), um basierend auf der einen oder den mehreren Messungen und einer Ausgabe einer Ausführung des einen oder der mehreren Provisioning-Code-Segmente (108) zu bestimmen (806), ob die eine oder die mehreren generierten Messungen die eine oder die mehreren Richtlinien (106) erfüllen, wobei die Ausführung des einen oder der mehreren sich außerhalb des verplombten Containers befindenden Provisioning-Code-Segmente durch die Vorrichtung (124) die Ausgabe generiert; und den verplombten Container (110) als Reaktion auf ein Bestimmen, dass die eine oder die mehreren generierten Messungen und die Ausgabe der Ausführung die eine oder die mehreren in der Nutzlast (104) empfangenen Richtlinien (106) erfüllen, an einem Trusted-Plattform-Modul (TPM) (120) des Trusted-Computing-Managers (126) zu öffnen (808), wobei eine Erfüllung der einen oder der mehreren Richtlinien (106) eine Integrität der Vorrichtung (124) bestätigt.

2. Vorrichtung (124) nach Anspruch 1, wobei der verplombte Container (110) und die eine oder die mehreren Richtlinien (106) durch einen Hash-Wert und einen Hash-Algorithmus geschützt sind, wobei der Hash-Wert unter Verwendung eines öffentlichen Schlüssels, der mit einem privaten Schlüssel verbunden ist, der in dem Trusted-Plattform-Modul (TPM) (120) der Vorrichtung (124) gespeichert ist, verschlüsselt ist, wobei der Trusted-Computing-Manager (126) weiter ausführbar ist, um den Hash-Wert unter Verwendung des privaten Schlüssels, der auf dem TPM (120) gespeichert ist und mit dem öffentlichen Schlüssel verbunden ist, der verwendet wird, um den Hash-Wert zu verschlüsseln, zu entschlüsseln, die eine oder die mehreren Richtlinien (106) und den verplombten Container (110) zu hashen, um einen neuen Hash-Wert zu generieren; den Hash-Wert mit dem neuen Hash-Wert zu vergleichen, um eine Integrität des verplombten Containers (110) und der einen oder der mehreren Richtlinien (106) zu bestimmen.

3. Vorrichtung (124) nach Anspruch 1, wobei der verplombte Container (110) und das eine oder die mehreren Provisioning-Code-Segmente unter Verwendung eines öffentlichen Schlüssels, der mit einem in dem Trusted-Plattform-Modul (TPM) (120) gespeicherten privaten Schlüssel verbunden ist, verschlüsselt werden, wobei der Trusted-Computing-Manager (126) weiter ausführbar ist, um den verplombten Container (110) und das eine oder die mehreren Provisioning-Code-Segmente unter Verwendung des in dem TPM (120) gespeicherten privaten Schlüssels zu entschlüsseln.

4. Vorrichtung (124) nach Anspruch 1, wobei das eine oder die mehreren Provisioning-Code-Segmente (108) aktive Richtlinien (410) sind, die von der Vorrichtung (124) ausführbar sind, um die eine oder die mehreren Richtlinien (106) zu erfüllen.

5. Vorrichtung (124) nach Anspruch 1, wobei der Trusted-Computing-Manager ein Firmware-basiertes Trusted-Plattform-Modul (TPM) (120) einschließt, das in einer Trusted-Execution-Umgebung (TEE) (322) der Vorrichtung (124) ausgeführt wird.

6. Vorrichtung (124) nach Anspruch 1, wobei die eine oder die mehreren Richtlinien (106) weiter auf einer oder mehreren geplanten Messungen von auf der Vorrichtung (124) ausführbarem Code basieren, wobei der auf der Vorrichtung (124) ausführbare Code vor dem Empfangen der Nutzlast (104) auf der Vorrichtung (124) gespeichert wird.

7. Verfahren (880) zum Bestätigen einer Integrität einer Vorrichtung (124), umfassend:

   Empfangen (802), an der Vorrichtung (124), einer Nutzlast (104), die einen verplombten Container (110), ein oder mehrere sich außerhalb des verplombten Containers (110) befindende

Provisioning-Code-Segmente (108) und eine oder mehrere sich außerhalb des verplombten Containers (110) befindende Richtlinien (106) enthält, wobei die eine oder die mehreren Richtlinien (106) auf dem einen oder den mehreren Provisioning-Code-Segmenten (108) basieren und mit dem verplombten Container (110) übereinstimmen;
Generieren (804), an der Vorrichtung (124), einer oder mehrerer Messungen des einen oder der mehreren Provisioning-Code-Segmente (108) basierend auf der einen oder den mehreren Richtlinien (106);
Ausführen des einen oder der mehreren sich außerhalb des verplombten Containers (110) befindenden Provisioning-Code-Segmente (108) durch die Vorrichtung (124), um eine Ausgabe zu generieren; und
Öffnen (808), an einem Trusted-Plattform-Modul (TPM) (120) eines Trusted-Computing-Managers (126) der Vorrichtung (124), des verplombten Containers (110) für die Vorrichtung (124) als Reaktion auf ein Bestimmen (806), dass die eine oder die mehreren generierten Messungen und die generierte Ausgabe die eine oder die mehreren Richtlinien (106) erfüllen, wobei die Erfüllung der einen oder der mehreren Richtlinien (106) eine Integrität der Vorrichtung (124) bestätigt.

8. Verfahren (800) nach Anspruch 7, wobei der verplombte Container (110) und die eine oder die mehreren Richtlinien (106) durch einen Hash-Wert und einen Hash-Algorithmus geschützt werden, wobei der Hash-Wert unter Verwendung eines öffentlichen Schlüssels, der mit einem privaten Schlüssel verbunden ist, der in dem Trusted-Plattform-Modul (TPM) (120) der Vorrichtung (124) gespeichert ist, verschlüsselt wird, wobei das Verfahren (800) weiter umfasst:

Entschlüsseln (504) des Hash-Werts unter Verwendung des privaten Schlüssels, der auf dem TPM (120) gespeichert ist und mit dem öffentlichen Schlüssel, der verwendet wird, um den Hash-Wert zu verschlüsseln, verbunden ist;
Hashen (506) der einen oder der mehreren Richtlinien (106) und des verplombten Containers (110), um einen neuen Hash-Wert zu generieren; und
Vergleichen (508) des Hash-Werts mit dem neuen Hash-Wert, um eine Integrität des verplombten Containers (110) und der einen oder der mehreren Richtlinien (106) zu bestimmen.

9. Verfahren (800) nach Anspruch 7, wobei der verplombte Container (110) und das eine oder die mehreren Provisioning-Code-Segmente (108) unter Verwendung eines öffentlichen Schlüssels, der mit einem in dem Trusted-Plattform-Modul (TPM) (120) der Vorrichtung (124) gespeicherten privaten Schlüssel verbunden ist, verschlüsselt werden, wobei das Verfahren (800) weiter umfasst:
Entschlüsseln (604) des verplombten Containers (110) und des einen oder der mehreren Provisioning-Code-Segmente (108) unter Verwendung des in dem TPM (120) gespeicherten privaten Schlüssels.

10. Verfahren (800) nach Anspruch 8, wobei das eine oder die mehreren Provisioning-Code-Segmente (108) aktive Richtlinien (410) sind, die von der Vorrichtung (124) ausführbar sind, um die eine oder die mehreren Richtlinien (106) zu erfüllen.

11. Verfahren (800) nach Anspruch 7, wobei die Vorrichtung (124) ein Firmware-basiertes Plattform-Modul (TPM) (120) einschließt, das in einer Trusted-Execution-Umgebung (322) der Vorrichtung (124) ausgeführt wird.

12. Verfahren (800) nach Anspruch 7, wobei die eine oder die mehreren Richtlinien (106) weiter auf einer oder mehreren geplanten Messungen von auf der Vorrichtung (124) ausführbarem Code basieren, wobei der auf der Vorrichtung (124) ausführbare Code vor dem Empfangen der Nutzlast (104) auf der Vorrichtung gespeichert wird.

13. Ein oder mehrere materielle prozessorlesbare Speichermedien (904), enthaltend Anweisungen zum Ausführen, auf einem oder mehreren Prozessoren (902) und Schaltkreisen einer Vorrichtung (124), eines Prozesses (800), umfassend:

Empfangen (802) einer Nutzlast (104), die einen verplombten Container (110), ein oder mehrere sich außerhalb des verplombten Containers (110) befindende Provisioning-Code-Segmente (108) und eine oder mehrere sich außerhalb des verplombten Containers befindende Richtlinien (106) enthält, wobei die eine oder die mehreren Richtlinien (106) auf dem einen oder den mehreren Provisioning-Code-Segmenten (108) basieren und mit dem verplombten Container (110) übereinstimmen;
Generieren (804) einer oder mehrerer Messungen des einen oder der mehreren Provisioning-Code-Segmente (108) basierend auf der einen oder den mehreren Richtlinien (106);
Ausführen des einen oder der mehreren sich außerhalb des verplombten Containers (110) befindenden Provisioning-Code-Segmente (108) durch die Vorrichtung (124), um eine Ausgabe zu generieren; und
Öffnen (808), an einem Trusted-Plattform-Modul (TPM) (120) eines Trusted-Computing-Ma-

nagers (126) der Vorrichtung (124), des verplombten Containers (110) für die Vorrichtung (124) als Reaktion auf ein Bestimmen (806) durch das TPM (120), dass die eine oder die mehreren generierten Messungen die eine oder die mehreren Richtlinien (106) erfüllen, basierend auf einer Ausgabe einer Ausführung durch das eine oder die mehreren Provisioning-Code-Segmente (108), wobei die Erfüllung der einen oder der mehreren Richtlinien (106) eine Integrität der Vorrichtung (124) bestätigt.

14. Das eine oder die mehreren materiellen prozessorlesbaren Speichermedien (904) nach Anspruch 13, wobei der verplombte Container (110) und die eine oder die mehreren Richtlinien (1060) durch einen Hash-Wert und einen Hash-Algorithmus geschützt sind, wobei der Hash-Wert unter Verwendung eines öffentlichen Schlüssels, der mit einem privaten Schlüssel verbunden ist, der in dem Trusted-Plattform-Modul (TPM) (120) der Vorrichtung gespeichert ist, verschlüsselt ist, wobei der Prozess weiter umfasst:

Entschlüsseln (504) des Hash-Werts unter Verwendung des privaten Schlüssels, der auf dem TPM (120) gespeichert ist und mit dem öffentlichen Schlüssel, der verwendet wird, um den Hash-Wert zu verschlüsseln, verbunden ist;
Hashen (506) der einen oder der mehreren Richtlinien (106) und des verplombten Containers (110), um einen neuen Hash-Wert zu generieren; und
Vergleichen (508) des Hash-Werts mit dem neuen Hash-Wert, um eine Integrität des verplombten Containers (110) und der einen oder der mehreren Richtlinien (106) zu bestimmen.

15. Das eine oder die mehreren materiellen prozessorlesbaren Speichermedien (904) nach Anspruch 13, wobei der verplombte Container (110) und das eine oder die mehreren Provisioning-Code-Segmente (108) unter Verwendung eines öffentlichen Schlüssels, der mit einem privaten Schlüssel verbunden ist, der in dem Trusted-Plattform-Modul (TPM) (120) der Vorrichtung (124) gespeichert ist, verschlüsselt werden, wobei der Prozess weiter umfasst:
Entschlüsseln (604) des verplombten Containers (110) und des einen oder der mehreren Provisioning-Code-Segmente (108) unter Verwendung des in dem TPM (120) gespeicherten privaten Schlüssels.

Revendications

1. Dispositif (124) comprenant :

un ou plusieurs processeurs (902) ;
une interface de charge utile (128) pouvant être exécutée par les un ou plusieurs processeurs (902) pour recevoir une charge utile (104) contenant un récipient scellé (110), un ou plusieurs segments de code d'approvisionnement (108) à l'extérieur du récipient scellé (110) et une ou plusieurs politiques (106) situées à l'extérieur du récipient scellé (110), les une ou plusieurs politiques (106) étant basées sur les un ou plusieurs segments de code d'approvisionnement (108) et correspondant au récipient scellé (110) ; et
un gestionnaire informatique de confiance (126) pouvant être exécuté par les un ou plusieurs processeurs (902) :
pour générer (804) une ou plusieurs mesures des un ou plusieurs segments de code d'approvisionnement (108) selon les une ou plusieurs politiques (106), pour déterminer (806) si les une ou plusieurs mesures générées satisfont les une ou plusieurs politiques (106) sur la base des une ou plusieurs mesures et d'une sortie d'exécution des un ou plusieurs segments de code d'approvisionnement (108), dans lequel l'exécution des un ou plusieurs segments de code d'approvisionnement situés à l'extérieur du récipient scellé par le dispositif (124) génère la sortie; et pour desceller (808), au niveau d'un module de plateforme de confiance (TPM) (120) du gestionnaire informatique de confiance (126), le récipient scellé (110) à la suite de la détermination que les une ou plusieurs mesures générées et la sortie d'exécution satisfont les une ou plusieurs politiques (106) reçues dans la charge utile (106), la satisfaction des une ou plusieurs politiques (106) confirmant l'intégrité du dispositif (124).

2. Dispositif (124) selon la revendication 1, dans lequel le récipient scellé (110) et les une ou plusieurs politiques (106) sont protégés par une valeur de hachage et un algorithme de hachage, la valeur de hachage étant cryptée à l'aide d'une clé publique associée à une clé privée stockée dans le module de plateforme de confiance (TPM) (120) du dispositif (124), le gestionnaire informatique de confiance (126) pouvant en outre être exécuté pour décrypter la valeur de hachage à l'aide de la clé privée stockée sur le module TPM (120) et associée à la clé publique utilisée pour crypter la valeur de hachage, les une ou plusieurs politiques (106) et le récipient scellé (110) pour générer une nouvelle valeur de hachage ; pour comparer la valeur de hachage avec la nouvelle valeur de hachage pour déterminer l'intégrité du récipient scellé (110) et des une ou plusieurs politiques (106).

3. Dispositif (124) selon la revendication 1, dans lequel le récipient scellé (110) et les un ou plusieurs segments de code d'approvisionnement sont cryptés à l'aide d'une clé publique associée à une clé privée stockée dans le module de plate-forme de confiance (TPM) (120), le gestionnaire informatique de confiance (126) pouvant en outre être exécuté pour décrypter le récipient scellé (110) et les un ou plusieurs segments de code d'approvisionnement à l'aide de la clé privée stockée dans le module TPM (120).

4. Dispositif (124) selon la revendication 1, dans lequel les un ou plusieurs segments de code d'approvisionnement (108) sont des politiques actives (410) pouvant être exécutées par le dispositif (124) pour satisfaire les une ou plusieurs politiques (106).

5. Dispositif (124) selon la revendication 1, dans lequel le gestionnaire informatique de confiance inclut un module de plate-forme de confiance faisant appel à un micrologiciel (TPM) (120) exécuté dans un environnement d'exécution de confiance (TEE) (322) du dispositif (124).

6. Dispositif (124) selon la revendication 1, dans lequel les une ou plusieurs politiques (106) sont en outre basées sur une ou plusieurs mesures projetées d'un code pouvant être exécuté sur le dispositif (124), le code pouvant être exécuté sur le dispositif (124) étant stocké sur le dispositif (124) avant de recevoir la charge utile (104).

7. Procédé (880) de confirmation de l'intégrité d'un dispositif (124) comprenant :

la réception (802), au niveau du dispositif (124), d'une charge utile (104) contenant un récipient scellé (110), un ou plusieurs segments de code d'approvisionnement (108) situés à l'extérieur du récipient scellé (110) et une ou plusieurs politiques (106) situées à l'extérieur du récipient scellé (110), les une ou plusieurs politiques (106) étant basées sur les un ou plusieurs segments de code d'approvisionnement (108) et correspondant au récipient scellé (110) ;
la génération (804), au niveau du dispositif (124), d'une ou de plusieurs mesures des un ou plusieurs segments de code d'approvisionnement (108) sur la base des une ou plusieurs politiques (106) ;
l'exécution des un ou plusieurs segments de code d'approvisionnement (108) situés à l'extérieur du récipient scellé (110) par le dispositif (124) pour générer une sortie ; et
le descellement (808), au niveau d'un module de plate-forme de confiance (TPM) (120) d'un gestionnaire informatique de confiance (126) du dispositif (124), du récipient scellé (110) pour le dispositif (124) à la suite de la détermination (806) que les une ou plusieurs mesures générées et la sortie générée satisfont les une ou plusieurs politiques (106), la satisfaction des une ou plusieurs politiques (106) confirmant l'intégrité du dispositif (124).

8. Procédé (800) selon la revendication 7, dans lequel le récipient scellé (110) et les une ou plusieurs politiques (106) sont protégés par une valeur de hachage et un algorithme de hachage, la valeur de hachage étant cryptée à l'aide d'une clé publique associée à une clé privée stockée dans le module de plate-forme de confiance (TPM) (120) du dispositif (124), le procédé (800) comprenant en outre :

le décryptage (504) de la valeur de hachage à l'aide de la clé privée stockée sur le module TPM (120) et associée à la clé publique utilisée pour crypter la valeur de hachage ;
le hachage (506) des une ou plusieurs politiques (106) et du récipient scellé (110) pour générer une nouvelle valeur de hachage ; et
la comparaison (508) de la valeur de hachage avec la nouvelle valeur de hachage pour déterminer l'intégrité du récipient scellé (110) et des une ou plusieurs politiques (106).

9. Procédé (800) selon la revendication 7, dans lequel le récipient scellé (110) et les un ou plusieurs segments de code d'approvisionnement (108) sont cryptés à l'aide d'une clé publique associée à une clé privée stockée dans le module de plate-forme de confiance (TPM) (120) du dispositif (124), le procédé (800) comprenant en outre :
le décryptage (604) du récipient scellé (110) et des un ou plusieurs segments de code d'approvisionnement (108) à l'aide de la clé privée stockée dans le module TPM (120).

10. Procédé (800) selon la revendication 8, dans lequel les un ou plusieurs segments de code d'approvisionnement (108) sont des politiques actives (410) pouvant être exécutées par le dispositif (124) pour satisfaire les une ou plusieurs politiques (106).

11. Procédé (800) selon la revendication 7, dans lequel le dispositif (124) inclut un module de plate-forme de confiance faisant appel à un micrologiciel (TPM) (120) exécuté dans un environnement d'exécution de confiance (322) du dispositif (124).

12. Procédé (800) selon la revendication 7, dans lequel les une ou plusieurs politiques (106) sont en outre basées sur une ou plusieurs mesures projetées d'un code pouvant être exécuté sur le dispositif (124), le code pouvant être exécuté sur le dispositif (124) étant stocké sur le dispositif avant de recevoir la charge utile (104).

13. Un ou plusieurs supports de stockage tangibles lisibles par un processeur (904) intégrés avec des instructions pour une exécution sur un ou plusieurs processeurs (902) et des circuits d'un dispositif (124), d'un processus (800) comprenant :

la réception (802) d'une charge utile (104) contenant un récipient scellé (110), un ou plusieurs segments de code d'approvisionnement (108) situés à l'extérieur du récipient scellé (110) et une ou plusieurs politiques (106) situées à l'extérieur du récipient scellé, les une ou plusieurs politiques (106) étant basées sur les un ou plusieurs segments de code d'approvisionnement (108) et correspondant au récipient scellé (110) ;
la génération (804) d'une ou de plusieurs mesures des un ou plusieurs segments de code d'approvisionnement (108) sur la base des une ou plusieurs politiques (106) ;
l'exécution des un ou plusieurs segments de code d'approvisionnement (108) situés à l'extérieur du récipient scellé (110) par le dispositif (124) pour générer une sortie ; et
le descellement (808), au niveau d'un module de plate-forme de confiance (TPM) (120) d'un gestionnaire informatique de confiance (126) du dispositif (124), du récipient scellé (110) pour le dispositif (124) à la suite de la détermination (806) par le module TPM (120) que les une ou plusieurs mesures générées satisfont les une ou plusieurs politiques (106) sur la base d'une sortie d'exécution par les un ou plusieurs segments de code d'approvisionnement (108), la satisfaction des une ou plusieurs politiques (106) confirmant l'intégrité du dispositif (124).

14. Un ou plusieurs supports de stockage tangibles lisibles par un processeur (904) selon la revendication 13, dans lequel le récipient scellé (110) et les une ou plusieurs politiques (1060) sont protégés par une valeur de hachage et un algorithme de hachage, la valeur de hachage étant cryptée à l'aide d'une clé publique associée à une clé privée stockée dans le module de plate-forme de confiance (TPM) (120) du dispositif, le processus comprenant en outre :

le décryptage (504) de la valeur de hachage à l'aide de la clé privée stockée sur le module TPM (120) et associée à la clé publique utilisée pour crypter la valeur de hachage ;
le hachage (506) des une ou plusieurs politiques (106) et du récipient scellé (110) pour générer une nouvelle valeur de hachage ; et
la comparaison (508) de la valeur de hachage avec la nouvelle valeur de hachage pour déterminer l'intégrité du récipient scellé (110) et des une ou plusieurs politiques (106).

15. Un ou plusieurs supports de stockage tangibles lisibles par un processeur (904) selon la revendication 13, dans lequel le récipient scellé (110) et les un ou plusieurs segments de code d'approvisionnement (108) sont cryptés à l'aide d'une clé publique associée à une clé privée stockée dans le module de plate-forme de confiance (TPM) (120) du dispositif (124), le processus comprenant en outre :

le décryptage (604) du récipient scellé (110) et des un ou plusieurs segments de code d'approvisionnement (108) à l'aide de la clé privée stockée dans le module TPM (120).

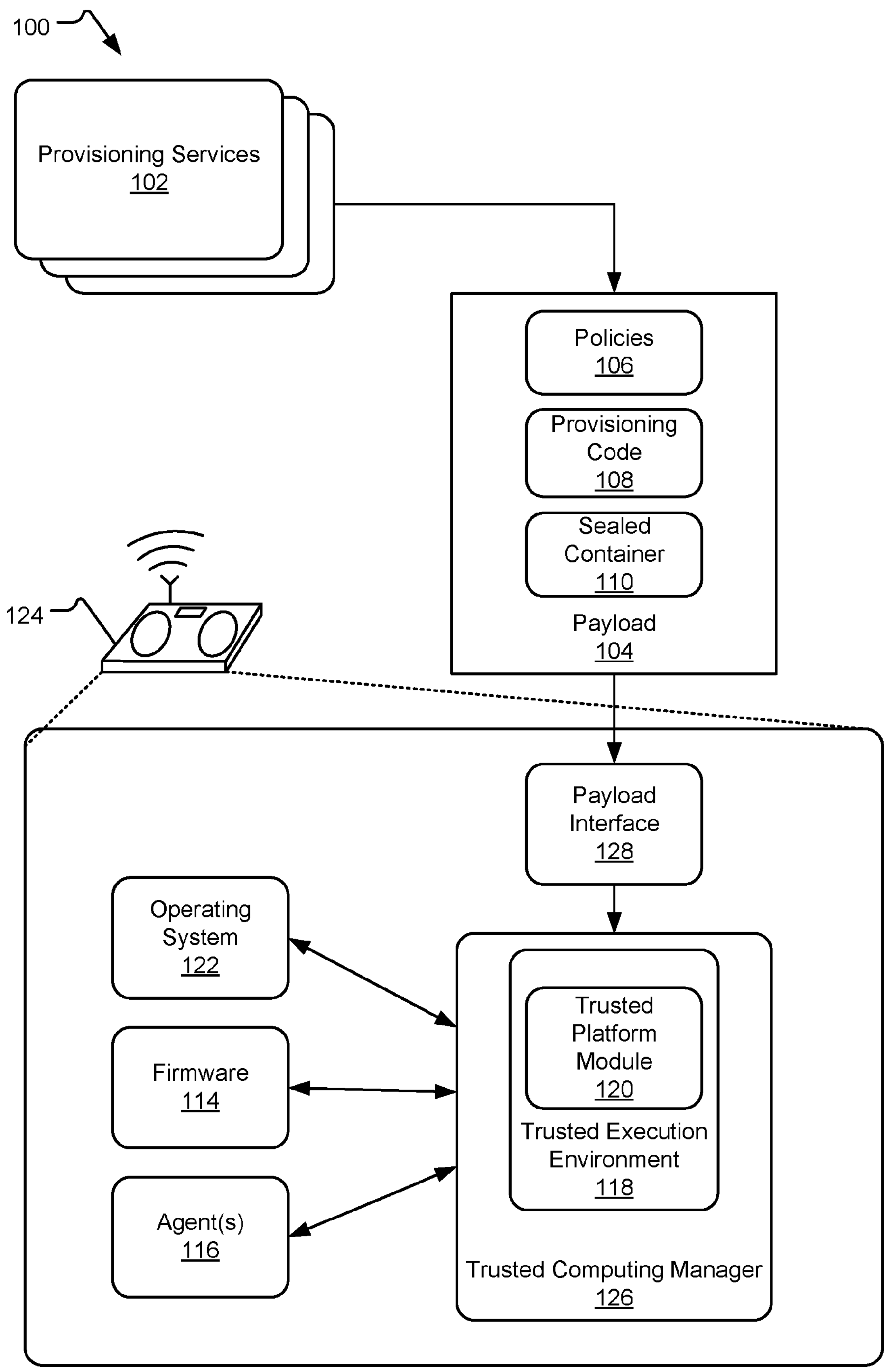
100
Provisioning Services
102
Policies
106
Provisioning Code
108
Sealed Container
110
Payload
104
124
Payload Interface
128
Operating System
122
Firmware
114
Agent(s)
116
Trusted Platform Module
120
Trusted Execution Environment
118
Trusted Computing Manager
126

FIG. 1

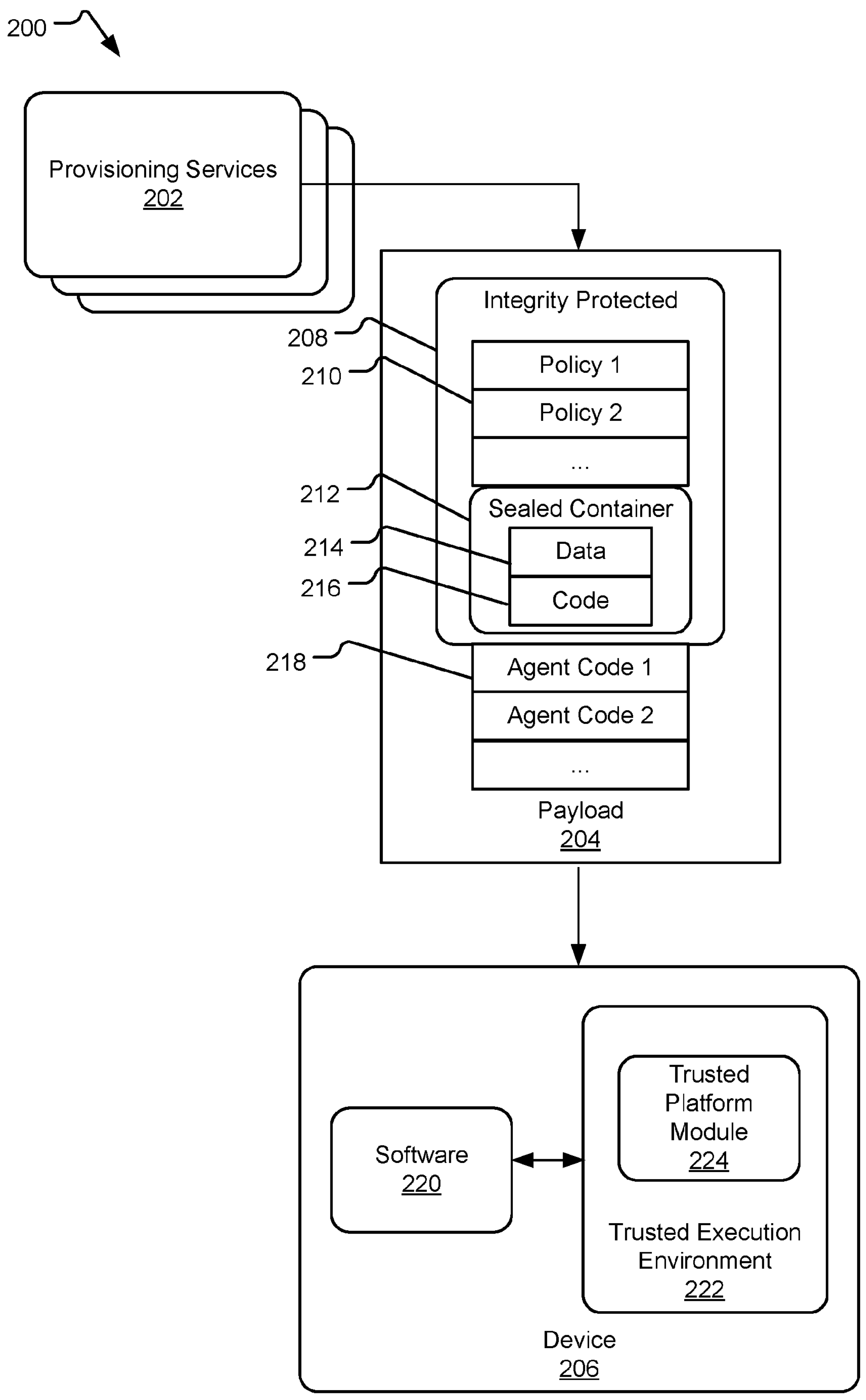
200
Provisioning Services
202
Integrity Protected
208
210
Policy 1
Policy 2
...
212
Sealed Container
214
Data
216
Code
218
Agent Code 1
Agent Code 2
...
Payload
204
Software
220
Trusted Platform Module
224
Trusted Execution Environment
222
Device
206

FIG. 2

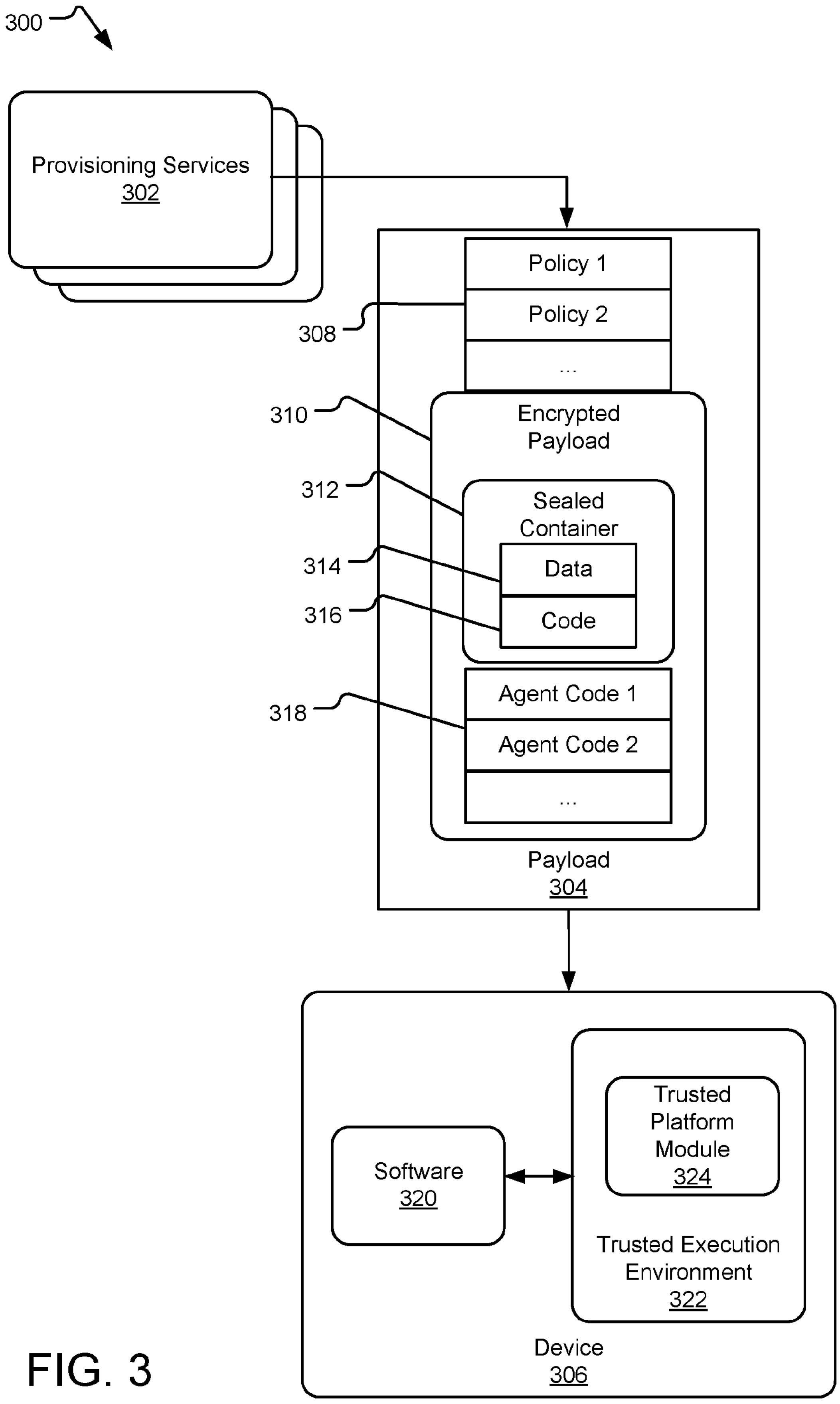
300
Provisioning Services
302
Policy 1
Policy 2
...
308
310
Encrypted
Payload
312
Sealed
Container
314
Data
316
Code
318
Agent Code 1
Agent Code 2
...
Payload
304
Software
320
Trusted
Platform
Module
324
Trusted Execution
Environment
322
Device
306

FIG. 3

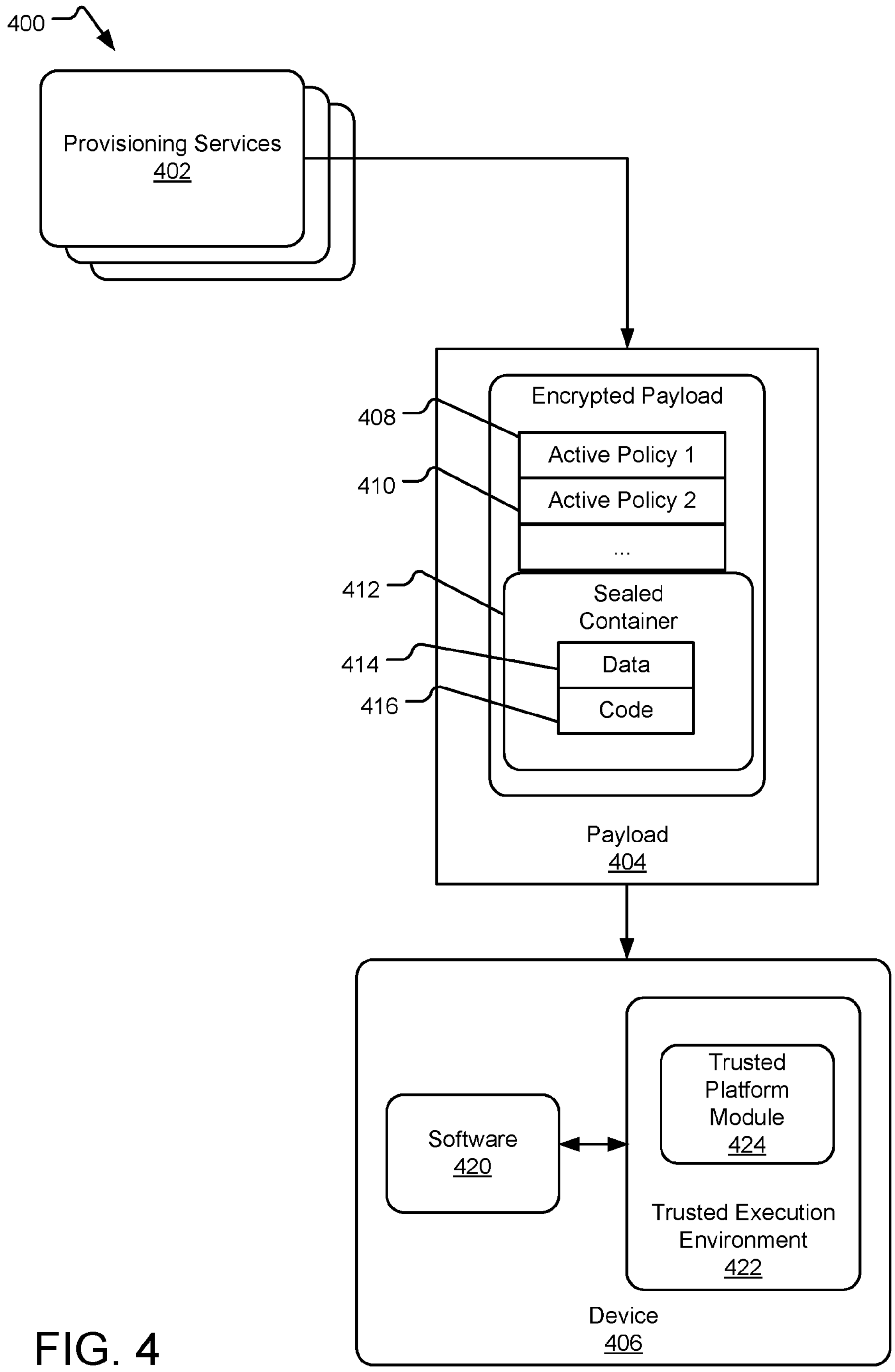
400
Provisioning Services
402
Encrypted Payload
408
Active Policy 1
410
Active Policy 2
...
412
Sealed
Container
414
Data
416
Code
Payload
404
Software
420
Trusted
Platform
Module
424
Trusted Execution
Environment
422
Device
406

FIG. 4

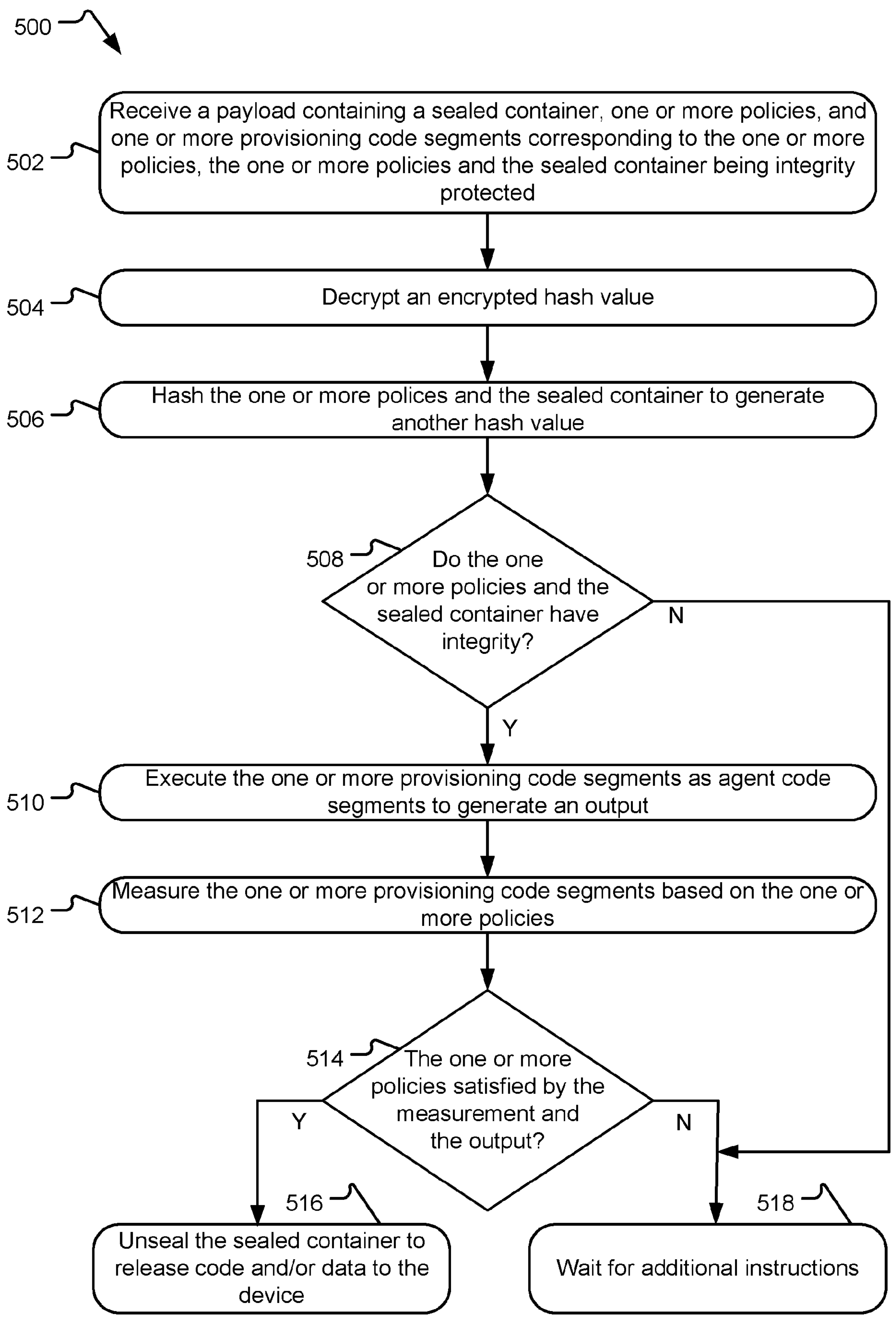
500
502
Receive a payload containing a sealed container, one or more policies, and one or more provisioning code segments corresponding to the one or more policies, the one or more policies and the sealed container being integrity protected
504
Decrypt an encrypted hash value
506
Hash the one or more polices and the sealed container to generate another hash value
508
Do the one or more policies and the sealed container have integrity?
N
Y
510
Execute the one or more provisioning code segments as agent code segments to generate an output
512
Measure the one or more provisioning code segments based on the one or more policies
514
The one or more policies satisfied by the measurement and the output?
Y
N
516
Unseal the sealed container to release code and/or data to the device
518
Wait for additional instructions

FIG. 5

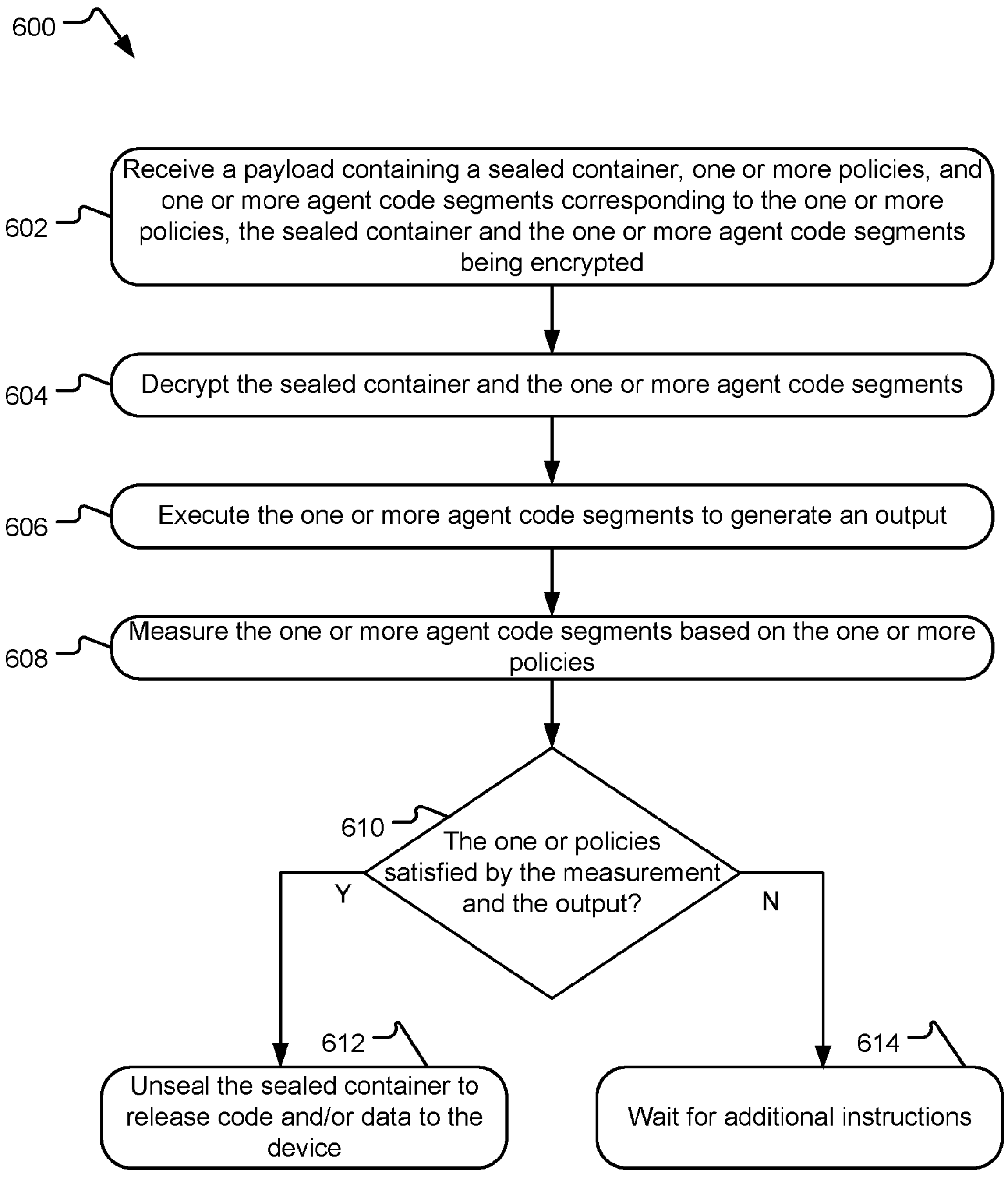
600
602
Receive a payload containing a sealed container, one or more policies, and one or more agent code segments corresponding to the one or more policies, the sealed container and the one or more agent code segments being encrypted
604
Decrypt the sealed container and the one or more agent code segments
606
Execute the one or more agent code segments to generate an output
608
Measure the one or more agent code segments based on the one or more policies
610
The one or policies satisfied by the measurement and the output?
Y
N
612
Unseal the sealed container to release code and/or data to the device
614
Wait for additional instructions

FIG. 6

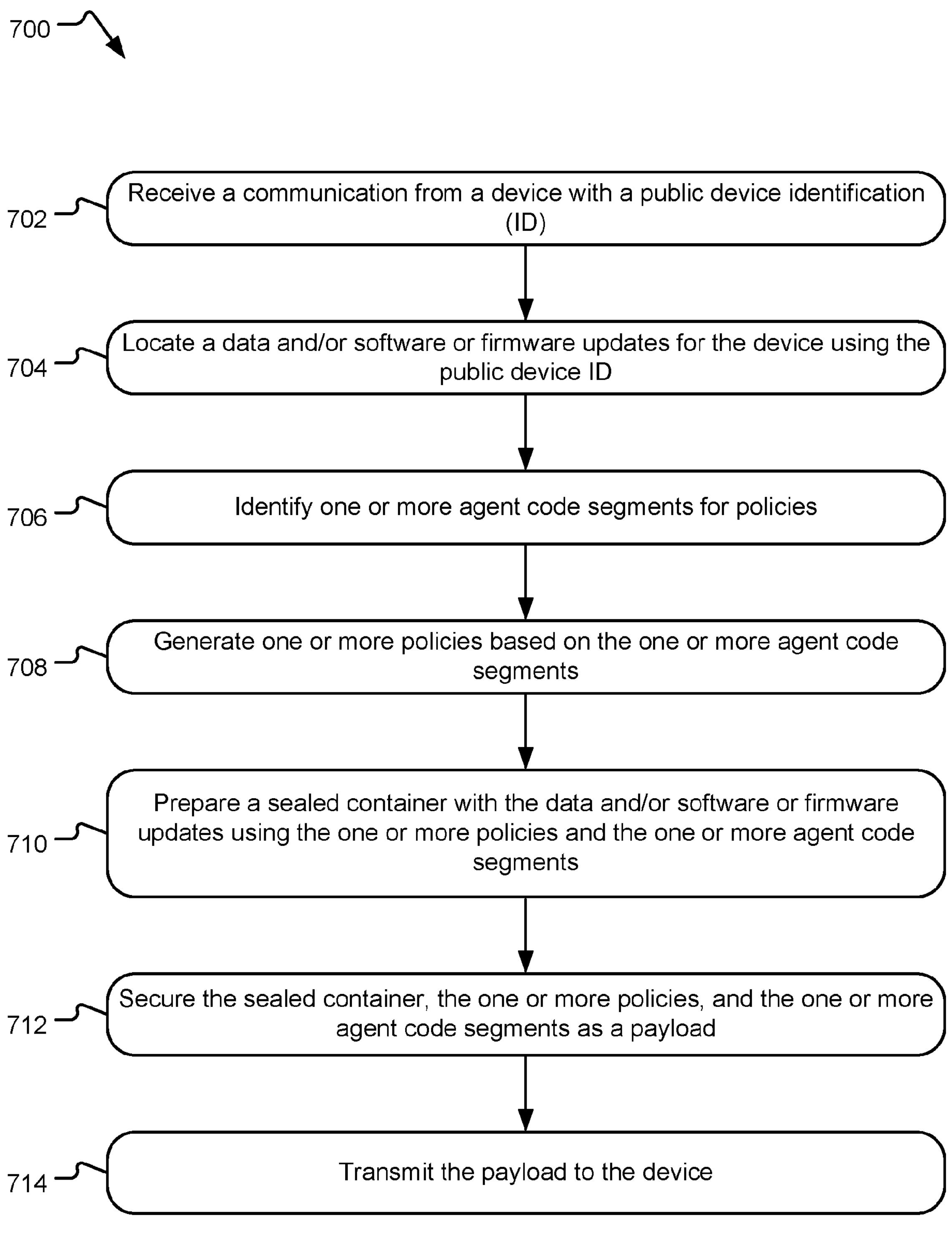
700
702
Receive a communication from a device with a public device identification (ID)
704
Locate a data and/or software or firmware updates for the device using the public device ID
706
Identify one or more agent code segments for policies
708
Generate one or more policies based on the one or more agent code segments
710
Prepare a sealed container with the data and/or software or firmware updates using the one or more policies and the one or more agent code segments
712
Secure the sealed container, the one or more policies, and the one or more agent code segments as a payload
714
Transmit the payload to the device

FIG. 7

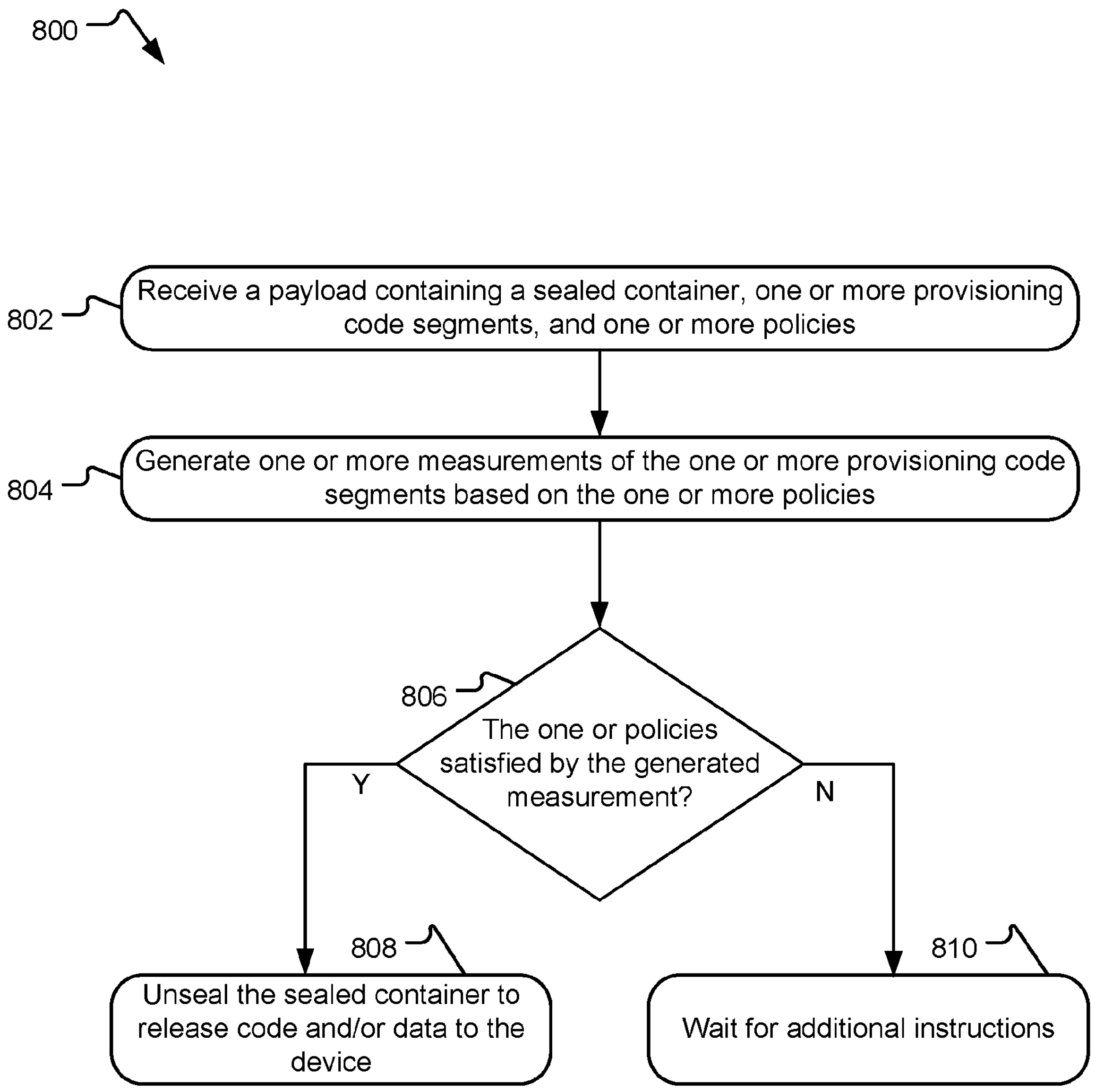
800
802
Receive a payload containing a sealed container, one or more provisioning code segments, and one or more policies
804
Generate one or more measurements of the one or more provisioning code segments based on the one or more policies
806
The one or policies satisfied by the generated measurement?
Y
N
808
Unseal the sealed container to release code and/or data to the device
810
Wait for additional instructions

FIG. 8

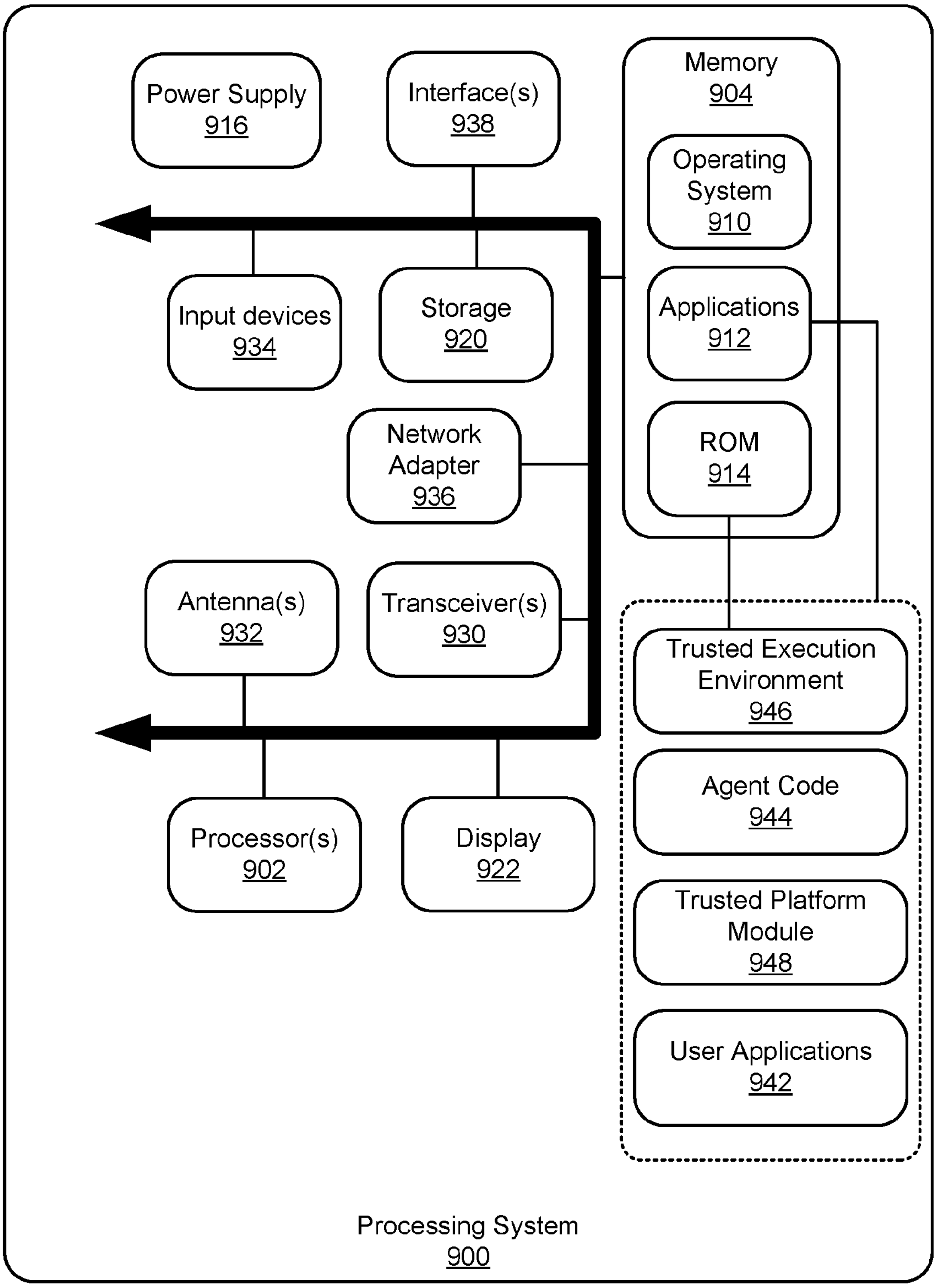
Power Supply 916
Interface(s) 938
Memory 904
Operating System 910
Applications 912
ROM 914
Input devices 934
Storage 920
Network Adapter 936
Antenna(s) 932
Transceiver(s) 930
Trusted Execution Environment 946
Agent Code 944
Trusted Platform Module 948
User Applications 942
Processor(s) 902
Display 922
Processing System 900

FIG. 9

REFERENCES CITED IN THE DESCRIPTION

Patent documents cited in the description

- US 2011179268 A1 **[0001]**
- US 2016054989 A1 **[0001]**
- US 2004127196 A1 **[0001]**