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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 October 2018 (25.10.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/194568 A1

(51) International Patent Classification:
G06F 9/06 (2006.01) *G06F 21/57* (2013.01)

(21) International Application Number:
PCT/US2017/028156

(22) International Filing Date:
18 April 2017 (18.04.2017)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).

(72) Inventors: **HUSSON, Remy**; Filton Road Stoke Gifford, Pt., Longdown Avenue, Stoke Gifford, Bristol Bristol BS34 8QZ (GB). **BALDWIN, Adrian**; Filton Road Stoke Gifford, Pt., Bristol Bristol BS34 8QZ (GB). **ELLAM, Daniel**; Filton Road Stoke Gifford, Pt., Bristol Bristol BS34 8QZ (GB).

(74) Agent: **BURROWS, Sarah** et al.; HP Inc., 3390 E. Harmony Road, Mail Stop 35, 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, DJ, 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, KH, KN, KP, KR, KW, 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,

(54) Title: EXECUTING PROCESSES IN SEQUENCE

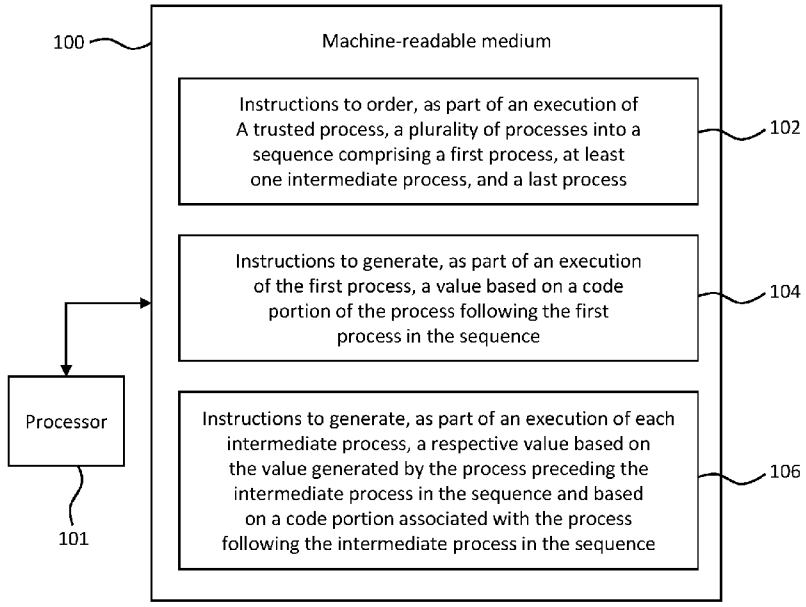

FIG. 1

(57) Abstract: In an example, a machine-readable medium includes instructions that, when executed by a processor, cause the processor to order, as part of an execution of a trusted process, a plurality of processes into a sequence comprising a first process, at least one intermediate process, and a last process. The machine-readable medium may further comprise instruction to cause the processor to generate, as part of an execution of the first process, a value based on a code portion of the process following the first process in the sequence, and to generate, as part of an execution of each intermediate process, a respective value based on the value generated by the process preceding the intermediate process in the sequence and based on a code portion associated with the process following the intermediate process in the sequence.

WO 2018/194568 A1

[Continued on next page]

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).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

Published:

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

EXECUTING PROCESSES IN SEQUENCE

BACKGROUND

[0001] A processing device may execute a plurality of processes. One or more processes may be trusted process, such as for example executing in a trusted zone of a processor, whereas other process may not be trusted processes. Malicious programs may attempt to modify a non-trusted process to change its operation. For example, a malicious program may attempt to modify a process to change the behaviour of the processing device or to intercept sensitive information.

BRIEF DESCRIPTION OF DRAWINGS

[0002] Examples will now be described, by way of non-limiting example, with reference to the accompanying drawings, in which:

[0003] Figure 1 is a simplified schematic of an example of a machine readable medium in association with a processor;

[0004] Figure 2 is a simplified schematic of an example of processes;

[0005] Figure 3 is a simplified schematic of another example of processes;

[0006] Figure 4 is a simplified schematic of a further example of processes;

[0007] Figure 5 is a simplified schematic of an example of a machine-readable medium in association with a processor;

[0008] Figure 6 is a simplified schematic of another example of a machine-readable medium in association with a processor;

[0009] Figure 7 is a simplified schematic of a further example of a machine-readable medium in association with a processor; and

[0010] Figure 8 is a simplified schematic of a processing device.

DETAILED DESCRIPTION

[0011] A processing device may execute a plurality of processes. At least one of the processes may be a trusted process. That is, for example, the trusted process is an

operating system process that is difficult for non-operating system processes to modify, and/or the process resides within a trusted execution environment of a processing device, such as for example ARM® TrustZone®. In some examples, the trusted process may be or reside within a hypervisor that executes beneath one or more virtual machines. In some examples, a trusted process may not be able to access memory associated with another process, though the trusted process may be able to exchange information and messages with the other process. In some examples, some processes may execute outside of the trusted execution environment, but may still be critical processes. For example, such processes may include operating system processes, security agents (for compliance or malware detection) and/or anti-virus processes.

[0012] In some examples, a first process may be able to access a portion of memory associated with a second process and may generate a value based on the memory. For example, the memory may contain at least a part of the other process. The value generated may provide a representation of the memory. For example, the value may be a hash value of the memory. The value or a representation of the value, such as a further value that incorporates the value, may be provided to another process (such as for example a trusted process) that verifies the value or representation. For example, the value or representation is compared to an expected value or representation. If the verification fails, then the second process may have been modified, for example by malicious code, malware, or malfunctioning code. The processing device may take appropriate action in this case, for example halting execution of the second process, halting execution of all processes, or shutting down.

[0013] In some examples, the value generated by the first process is based on an information element. For example, the value is a hash value of the information element and a portion of memory associated with the second process. The information element may be based on a portion of memory associated with the first process. For example, the information element may be a hash value of the portion of memory associated with the first process, and may be received from another process. In this scenario, the value generated by the first process incorporates a representation of code associated with the first process as well as a representation of code associated with the second process. This increases the difficulty of undetected tampering of processes by a malicious program for example, as the malicious program may need to modify several processes to avoid detection.

[0014] Figure 1 shows an example of a machine-readable medium 100 that may be, for example, a machine-readable medium including instructions for verifying the

integrity of a plurality of processes. The machine-readable medium 100 includes instructions 102 that, when executed by the processor 101, cause the processor 101 to order, as part of an execution of a trusted process, a plurality of processes into a sequence comprising a first process, at least one intermediate process, and a last process 106. In some examples, the order is chosen on in a random or pseudorandom manner. Choosing the order in a random manner may make it more difficult for a malicious or malfunctioning program to modify any of the first, intermediate and last processes, because a final value generated by the last process may be affected by the order of these processes.

[0015] The machine-readable medium 100 of Figure 1 also includes instructions 104 that, when executed by a processor, cause the processor to generate, as part of an execution of the first process, a value based on a code portion of the process following the first process in the sequence (that is, the first one of the at least one intermediate process 204). In some examples, the value may be a hash value that is generated by running a hash function that has as an input a portion of memory containing at least part of the process following the first process in the sequence. In some examples, the first process may commence generating a value in response to an event, such as for example receiving an instruction from the trusted process. In some examples, the trusted process may send repeat instructions to the first process in the sequence for checking of the processes on an ongoing basis, such as periodically, and/or in response to an event such as for example possible detection of a malicious program.

[0016] In some examples, the value generated by the first process is also based on a seed, such as for example a random value, a pseudorandom value or a timestamp. In some examples, the value is generated using a hash function on the portion of memory associated with the next process in the sequence and also on the seed, which is for example the first or last data provided to the hash function. As a result, the value generated by the first process may be different if the instructions 102 are repeated. This may make it more difficult for a malicious program to modify any of the processes to avoid detection. For example, without the seed, the malicious program may be modify the second process to receive the value from the first process and to replace it with the correct value. This action may be thwarted if the value is modified using a seed.

[0017] The machine-readable medium 100 further includes instructions 106 that, when executed by a processor, cause the processor 101 to generate, as part of an execution of each intermediate process, a respective value based on the value generated by the process preceding the intermediate process in the sequence and

based on a code portion associated with the process following the intermediate process in the sequence. As a result, for example, the value that is generated by the first process 202 is passed along the sequence of processes and 206, being modified into the respective value at each intermediate process. At the last intermediate process 204, the respective value (which may be referred to as the final value) is based on the process preceding the intermediate process in the sequence and based on a code portion associated with the last process. In some examples the final value is verified, for example by comparing the final value to an expected final value. The final value will differ from the expected final value if code associated with any of the processes has been modified, and hence such modification can be detected.

[0018] The value generated by a process may be provided to the next process in the sequence in some examples by providing the value directly to the process in a message, which may be for example a message instructing the next process to generate its own respective value, or for example by storing the value in a location in memory that is accessible by the next process. In some examples, where a process sends an instruction to the next process to generate its respective value, the process may determine which process is next by for example accessing a location in memory that provides this information. For example, the instructions 102 may include instructions such that the trusted process generates a table in memory that is accessible by the processes 202, 204 and 206 and indicates the sequence into which the processes are ordered. In some examples, the processes may be able to determine which is the next process. For example, each process may include a random or pseudorandom number generator that generates a random number to be used in determining the next process in the sequence. The first process may receive or retrieve a seed from the trusted process such that the seed determines the pseudorandom order of the processes.

[0019] In some examples, each process may be associated with a respective key. The instructions 104 and 106 may include instructions such that each process signs the value it generates with the appropriate key before the value is passed to the next process in the sequence. The keys may be generated by the trusted process and provided to or made accessible to each process, and may form part of a key pair, with the trusted process holding the other key of each pair such that the trusted process can verify the signatures. Using keys may make it more difficult for a malicious program to modify a process undetected, such as for example anticipate the value received from the preceding process in the sequence and replace it with a corrected value. In some examples, the keys may be modified by the trusted process between repeat executions

of the instructions within the machine-readable medium 100 to avoid the processes generating identical values in repeat executions.

[0020] In some examples, there may be only one intermediate process. In this case, the instructions 104 may comprise instructions that, when executed by a processor, cause the processor to generate, as part of an execution of the intermediate process 204, a respective value based on the value generated by the first process and based on a code portion associated with the last process.

[0021] Figure 2 shows an example of a trusted process 200, a first process 202, a second process 204 and a last process 206. The processes 202, 204 and 206 have been ordered into a sequence.

[0022] In some examples, the last process may be the first process (i.e. the processes may form a closed loop). Figure 3 shows an example of processes including a trusted process 300, a first process 302 and an intermediate process 304. The first process 302 is also the last process in the sequence. In this case, the instructions 104 may comprise instructions that, when executed by a processor, cause the processor to generate, as part of an execution of the intermediate process 204, a respective value based on the value generated by the first process and based on a code portion associated with the first process. The final value generated by the intermediate process 304 may be based on code portions associated with both the first process 302 and the intermediate process 304. In some examples, where the first process is also the last process, there may be multiple intermediate processes 304.

[0023] Figure 4 shows an example of processes including a trusted process 400 and a first process 402. Intermediate processes comprise a second process 404 and third process 406. In this example, the first process 402 is also the last process.

[0024] The trusted process 400 generates a seed 408 that is provided to a hash function 410 of the first process 402. The hash function 410 generates a value 414 based on the seed 408, a code portion associated with the second process 404, and a first key 412 associated with the first process 402. The value 414 is passed to the second process 404.

[0025] The second process 404 includes a hash function 416 that generates a value 418 based on the value 414, a code portion associated with the third process 406, and a key 420 associated with the second process 404. The value 418 is passed to the third process 406.

[0026] The third process 406 includes a hash function 422 that generates a value 424 based on the value 418, a code portion associated with the first process 402, and a key 426 associated with the third process 406. The value 424 is the final value that may be verified to determine the integrity of the first process 402, the second process 404 and the third process 406. If the final value 424 differs from an expected final value, this may indicate that at least one of the processes 402, 404 and 406 has been modified.

[0027] In some examples, the final value 424 is provided to the trusted process 400, such as for example being provided by the third process 406, or being provided to the first process 402 which forwards the final value to the trusted process 400. The trusted process may compare the final value 424 with an expected final value to verify the integrity of the processes 402, 404 and 406. In some examples, the trusted process 400 may generate the expected final value based on the seed 408, the keys 412, 420 and 426 and the expected output of the hash functions 410, 416 and 422, and may compare the expected final value to the received final value.

[0028] Figure 5 shows an example of a machine-readable medium 500 that may be, for example, a machine-readable medium including instructions for verifying the integrity of a plurality of processes. The machine-readable medium includes instructions 502, 504 and 506 that correspond to the instructions 102, 104 and 106 respectively shown in Figure 1. The machine-readable medium 500 also includes instructions 508 that, when executed by a processor 501, cause the processor to modify, as part of an execution of a last one of the intermediate processes, a final value based on the value generated by the process preceding the intermediate process in the sequence and based on a code portion associated with one of the first process and a further process. Hence, the “last” process in the sequence can be the first process or another process, and the final value is based on the code of this process.

[0029] The machine-readable medium 500 also includes instructions 510 that, when executed by a processor, cause the processor to receive, as part of an execution of the trusted process, a representation of the final value. The trusted process may compare this representation to an expected value to verify it. The representation of the final value may in some examples be the final value.

[0030] The machine-readable medium 500 also includes instructions 512 that, when executed by a processor 501, cause the processor to modify, as part of an execution of the first process, the value generated by the first process using a key associated with the first process, to modify, as part of an execution of each intermediate

process, the respective value generated by the intermediate process using a respective key associated with the intermediate process, and to modify, as part of an execution of the last process, the final value using a key associated with the last process. Therefore, as described above with reference to the example shown in Figure 4, the value can be "protected" by signing with keys as it gets passed between processes. As a result, for a malicious program to modify a process, it may need access to all of the keys as well as other information (such as for example the order of the processes) to be able to disguise any modifications to the process by replacing an incorrect value with a corrected one.

[0031] The machine-readable medium 500 also includes instructions 514 that, when executed by a processor 501, cause the processor to modify, as part of an execution of the first process, the value generated by the first process using a seed. The final value may then be based on the seed as well as other information (such as for example the code portions in memory of the processes) to ensure that the final value varies between repeat cycles of a procedure for determining process integrity, and the final value cannot be forged by for example an altered process.

[0032] The machine-readable medium 500 also includes instructions 516 that, when executed by a processor 501, cause the processor to pause execution of the process following the first process, calculate the value, and resume execution of the process following the first process. In this case, a process can create a value based on a portion of memory around which execution of the next process is paused. This may prevent a scenario in which a malicious program modifies a process and keeps an unmodified version of the process elsewhere in memory to ensure that the preceding process in the sequence generates the correct value from the unmodified code. By taking into account the location of execution of a process, it can be ensured that the executing version of a process is considered and not a copy of it. In some examples, the process being paused may always pause at the same point, such that the region of memory that contains that process may remain in the same place relative to the pause point. The process being paused may include code to ensure that the process pauses at the same point. The presence of a pause may be taken into account when determining a schedule for performing checks.

[0033] In some examples, a first process that checks the next process (that is, creates a value based on a portion of memory containing at least part of the next process) may send an interrupt signal to the next process to pause it, wait for the next process to be paused, create the value, and instruct the next process to perform the same actions in respect of its own next process. The first process may then resume

execution of the next process from the point at which it was paused, though this may occur after the next process has completed its actions in respect of its own next process. In some examples, all processes in the sequence may be paused before the first process starts checking its next process in the sequence, and all the processes may be resumed from their respective pause points once the final value has been created.

[0034] Figure 6 shows an example of a machine-readable medium 600 storing instructions that may be, for example, instructions executed by a processor 601 during verification of the integrity of a plurality of processes. The machine-readable medium 600 includes instructions 602 that, when executed by the processor 601, cause the processor to execute a first process to generate a value based on an information element and a code portion associated with a second process, the information element being based on a code portion associated with the first process. As such, the value is based on the code of both the first process and the second process. The machine-readable medium 600 also includes instructions 604 that, when executed by the processor 601, cause the processor to provide the value to one of the second process and a trusted process. Therefore, the second process may in some examples go on to create a further value that incorporates a representation of code of a third process, or the trusted process may verify the value from the first process.

[0035] Figure 7 shows an example of a machine-readable medium 700 storing instructions that may be, for example, instructions executed by a processor 701 during verification of the integrity of a plurality of processes. The machine-readable medium 700 includes instructions 702 that, when executed by a processor 701, cause the processor to execute a first process to generate a value based on an information element and a code portion associated with a process, the information element being based on a code portion associated with the first process. The first process may for example receive a value (the information element) based on its own code, modify it based on code of the second process, and forward the modified value to the second process or a trusted process. Modification of the first or second process may therefore in some examples be revealed by the modified value being different from the expected modified value.

[0036] The machine-readable medium also comprises instructions 704 that, when executed by a processor 701, cause the processor to provide the value generated by the first process to one of the second process and a trusted process. The second process may further modify the value, or the trusted process can verify the value.

[0037] The instructions 702 may comprise instructions 706 that, when executed by a processor 701, cause the processor to instruct the second process to pause execution, calculate the value, and instruct the second process to resume execution. As indicated above, the value may be based on a code portion of an executing process instead of an unmodified copy of that process.

[0038] The instructions 702 may comprise instructions 708 that, when executed by a processor 701, cause the processor to instruct the second process to calculate a hash function of a portion of memory that contains at least part of the second process. The value may then depend on a code portion of the second process and therefore any modification of the second process will be reflected in the value.

[0039] The machine-readable medium also comprises instructions 710 that, when executed by a processor, cause the processor to compare the value with an expected value. If the value differs from the expected value, this may indicate that the first process or the second process has been modified, for example by a malicious process. In some examples, action may subsequently be taken if the value is unexpected, such as terminating one or more processes, or pausing or shutting down a processing device that is executing instructions from the machine-readable medium 700.

[0040] The machine-readable medium also comprises instructions 712 that, when executed by a processor, cause the processor to execute the second process to receive the value, generate a further value based on the value and a code portion associated with a further process, and provide the further value to one of the further process and the trusted process. As a result, in some examples, the further value can be based on code portions associated with the first process, the second process and the further process, and modification of any of these processes may be revealed in the further value generated by the second process.

[0041] Figure 8 shows an example of a processing device 800 that may for example determine the integrity of running processes. The processing device 800 comprises a memory 802 and a plurality of ordered processing modules. The plurality of ordered processing modules are ordered into a sequence and include a first processing module 804 and a second processing module 806.

[0042] The first processing module 804 in the sequence includes a first hash calculation module 808 to calculate a hash value of a portion of the memory 802 associated with the next processing module. This may be the last processing module 806, or an intermediate processing module in some examples.

[0043] The last processing module 806 includes a last hash calculation module 810 to calculate a hash value from the hash value calculated by its preceding processing module (which may be the first processing module 804 or an intermediate processing module) and a hash value of a portion of the memory associated with the first processing module 802 or another processing module. A verification module 812 may verify the hash value calculated by the last hash calculation module 810. For example, the hash value is compared with a predicted hash value. If a discrepancy is detected, which may indicate that one of the processing modules has been modified, then the verification module may in some examples take action, such as shutting down the processing device 800, shutting down one or more of the processing modules, and/or presenting a message to a user of the processing device 800.

[0044] Examples in the present disclosure can be provided as methods, systems or machine readable instructions, such as any combination of software, hardware, firmware or the like. Such machine readable instructions may be included on a computer readable storage medium (including but is not limited to disc storage, CD-ROM, optical storage, etc.) having computer readable program codes therein or thereon.

[0045] The present disclosure is described with reference to flow charts and/or block diagrams of the method, devices and systems according to examples of the present disclosure. Although the flow diagrams described above show a specific order of execution, the order of execution may differ from that which is depicted. Blocks described in relation to one flow chart may be combined with those of another flow chart. It shall be understood that each flow and/or block in the flow charts and/or block diagrams, as well as combinations of the flows and/or diagrams in the flow charts and/or block diagrams can be realized by machine readable instructions.

[0046] The machine readable instructions may, for example, be executed by a general purpose computer, a special purpose computer, an embedded processor or processors of other programmable data processing devices to realize the functions described in the description and diagrams. In particular, a processor or processing apparatus may execute the machine readable instructions. Thus functional modules of the apparatus and devices may be implemented by a processor executing machine readable instructions stored in a memory, or a processor operating in accordance with instructions embedded in logic circuitry. The term ‘processor’ is to be interpreted broadly to include a CPU, processing unit, ASIC, logic unit, or programmable gate array etc. The methods and functional modules may all be performed by a single processor or divided amongst several processors.

[0047] Such machine readable instructions may also be stored in a computer readable storage that can guide the computer or other programmable data processing devices to operate in a specific mode.

[0048] Such machine readable instructions may also be loaded onto a computer or other programmable data processing devices, so that the computer or other programmable data processing devices perform a series of operations to produce computer-implemented processing, thus the instructions executed on the computer or other programmable devices realize functions specified by flow(s) in the flow charts and/or block(s) in the block diagrams.

[0049] Further, the teachings herein may be implemented in the form of a computer software product, the computer software product being stored in a storage medium and comprising a plurality of instructions for making a computer device implement the methods recited in the examples of the present disclosure.

[0050] While the method, apparatus and related aspects have been described with reference to certain examples, various modifications, changes, omissions, and substitutions can be made without departing from the spirit of the present disclosure. It is intended, therefore, that the method, apparatus and related aspects be limited only by the scope of the following claims and their equivalents. It should be noted that the above-mentioned examples illustrate rather than limit what is described herein, and that those skilled in the art will be able to design many alternative implementations without departing from the scope of the appended claims.

[0051] The word “comprising” does not exclude the presence of elements other than those listed in a claim, “a” or “an” does not exclude a plurality, and a single processor or other unit may fulfil the functions of several units recited in the claims.

[0052] The features of any dependent claim may be combined with the features of any of the independent claims or other dependent claims.

CLAIMS

1. A machine-readable medium comprising instructions that, when executed by a processor, cause the processor to:

order, as part of an execution of a trusted process, a plurality of processes into a sequence comprising a first process, at least one intermediate process, and a last process;

generate, as part of an execution of the first process, a value based on a code portion of the process following the first process in the sequence; and

generate, as part of an execution of each intermediate process, a respective value based on the value generated by the process preceding the intermediate process in the sequence and based on a code portion associated with the process following the intermediate process in the sequence.

2. The machine-readable medium of claim 1, comprising instructions that, when executed by a processor, cause the processor to modify, as part of an execution of a last one of the at least one intermediate process, a final value based on the value generated by a preceding process in the sequence and based on a code portion associated with one of the first process and a further process.

3. The machine-readable medium of claim 1, comprising instructions that, when executed by a processor, cause the processor to receive, as part of an execution of the trusted process, a representation of the final value.

4. The machine-readable medium of claim 1, wherein ordering the plurality of processes into a sequence comprises randomizing the order of the plurality of processes in the sequence.

5. The machine-readable medium of claim 1, comprising instructions that, when executed by a processor, cause the processor to:

modify, as part of an execution of the first process, the value generated by the first process using a key associated with the first process;

modify, as part of an execution of each intermediate process, the respective value generated by the intermediate process using a respective key associated with the intermediate process; and

modify, as part of an execution of the final process, the final value using a key associated with the last process.

6. The machine-readable medium of claim 1, comprising instructions that, when executed by a processor, cause the processor to modify, as part of an execution of the first process, the value generated by the first process using a seed.

7. The machine-readable medium of claim 1, comprising instructions that, when executed by a processor, cause the processor to verify the final value.

8. The machine-readable medium of claim 1, wherein the instructions to cause the processor to generate, as part of an execution of the first process, the value based on a code portion of a process following the first process in the sequence comprise instructions to cause the processor to pause execution of the process following the first process, calculating the value, and resuming execution of the process following the first process.

9. A machine-readable medium comprising instructions that, when executed by a processor, cause the processor to:

execute a first process to generate a value based on an information element and a code portion associated with a second process, the information element being based on a code portion associated with the first process; and

provide the value to one of the second process and a trusted process.

10. The machine-readable medium of claim 9, wherein the instructions to generate the value comprise instructions that, when executed by a processor, cause the processor to instruct the second process to pause execution, calculate the value, and instruct the second process to resume execution.

11. The machine-readable medium of claim 9, wherein the information element comprises at least one of a key, a seed and a value from another process.

12. The machine-readable medium of claim 9, wherein the instructions to generate the value comprise instructions that, when executed by a processor, cause the processor to calculate a hash function of a portion of memory that contains at least part of the second process.

13. The machine-readable medium of claim 9, comprising instructions that, when executed by a processor, cause the processor to compare the value with an expected value.

14. The machine-readable medium of claim 9, comprising instructions that, when executed by a processor, cause the processor to execute the second process to receive the value, generate a further value based on the value and a code portion associated with a further process, and provide the further value to one of the further process and the trusted process.

15. A device comprising:

a memory;

a plurality of ordered processing modules including a first processing module and a last processing module;

the first processing module including a first hash calculation module to calculate a hash value of a portion of the memory associated with the next processing module;

the last processing module including a last hash calculation module to calculate a hash value from the hash value calculated by its preceding processing module and a hash value of a portion of the memory associated with the first processing module or another processing module; and

a verification module to verify the hash value calculated by the last processing module.

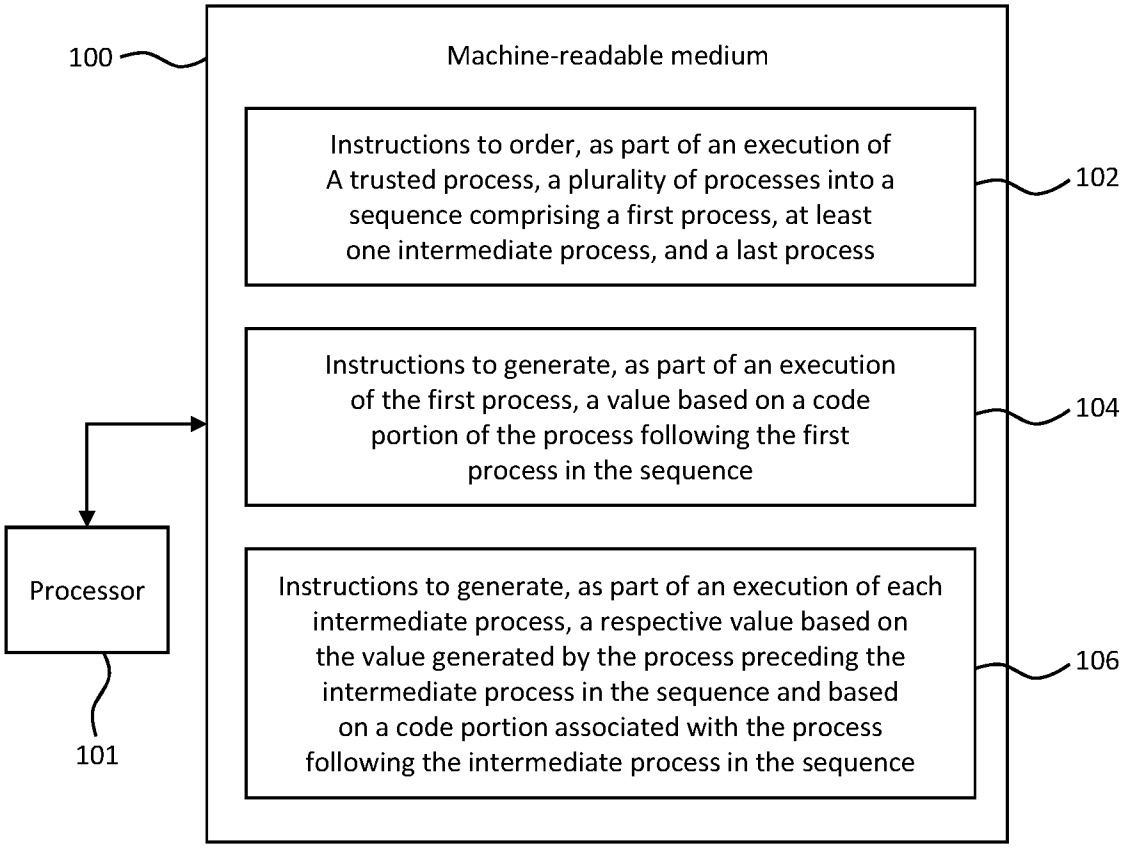
100
Machine-readable medium
Instructions to order, as part of an execution of A trusted process, a plurality of processes into a sequence comprising a first process, at least one intermediate process, and a last process
102
Instructions to generate, as part of an execution of the first process, a value based on a code portion of the process following the first process in the sequence
104
Processor
Instructions to generate, as part of an execution of each intermediate process, a respective value based on the value generated by the process preceding the intermediate process in the sequence and based on a code portion associated with the process following the intermediate process in the sequence
106
101

FIG. 1

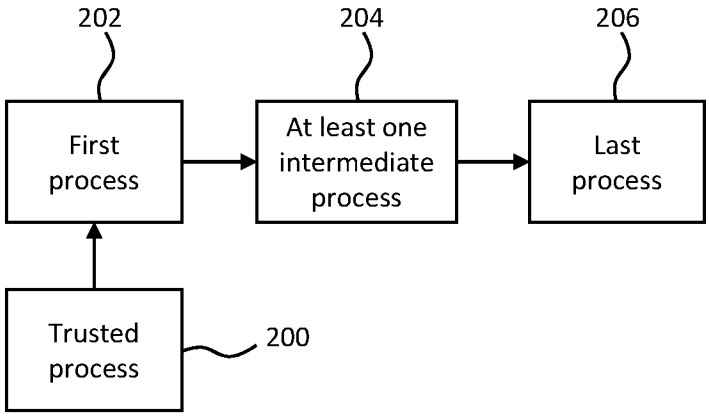
202
204
206
First process
At least one intermediate process
Last process
Trusted process
200

FIG. 2

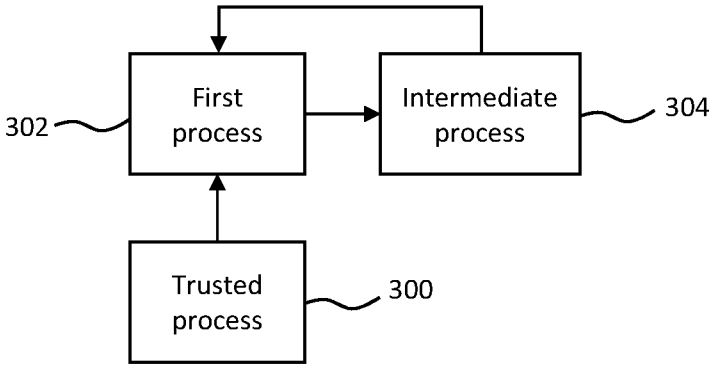
First process
Intermediate process
Trusted process
302
304
300

FIG. 3

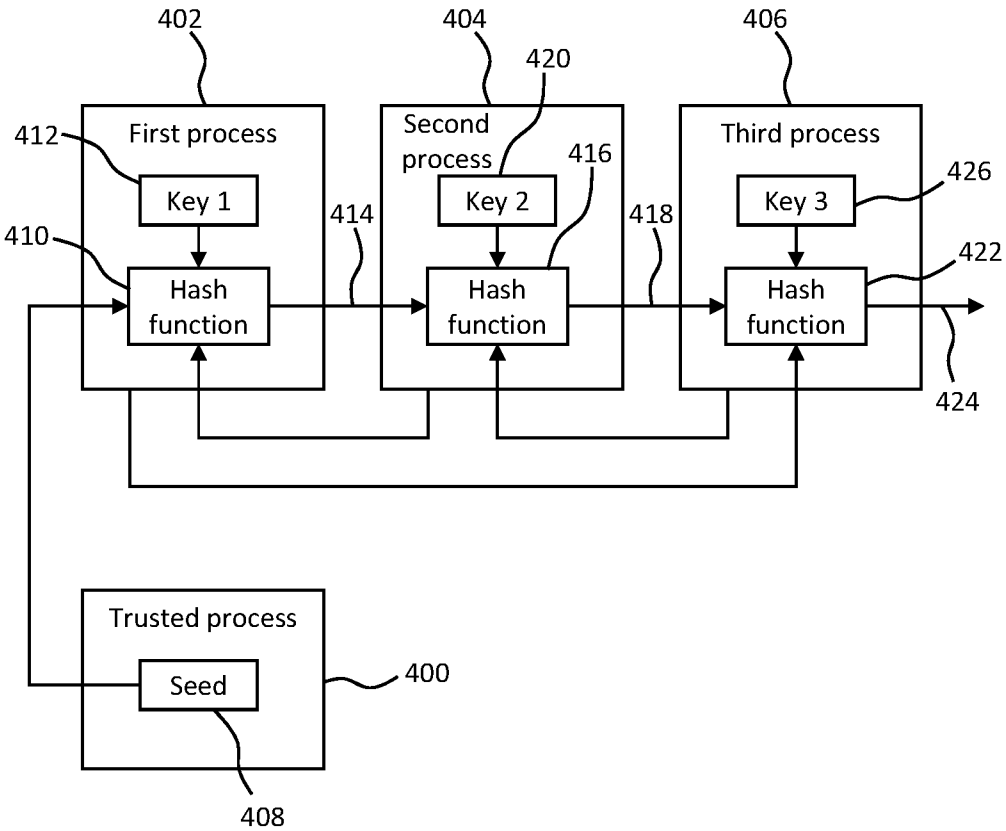
402
404
406
420
First process
Second process
Third process
412
Key 1
Key 2
Key 3
426
410
414
416
418
422
Hash function
Hash function
Hash function
424
Trusted process
Seed
400
408

FIG. 4

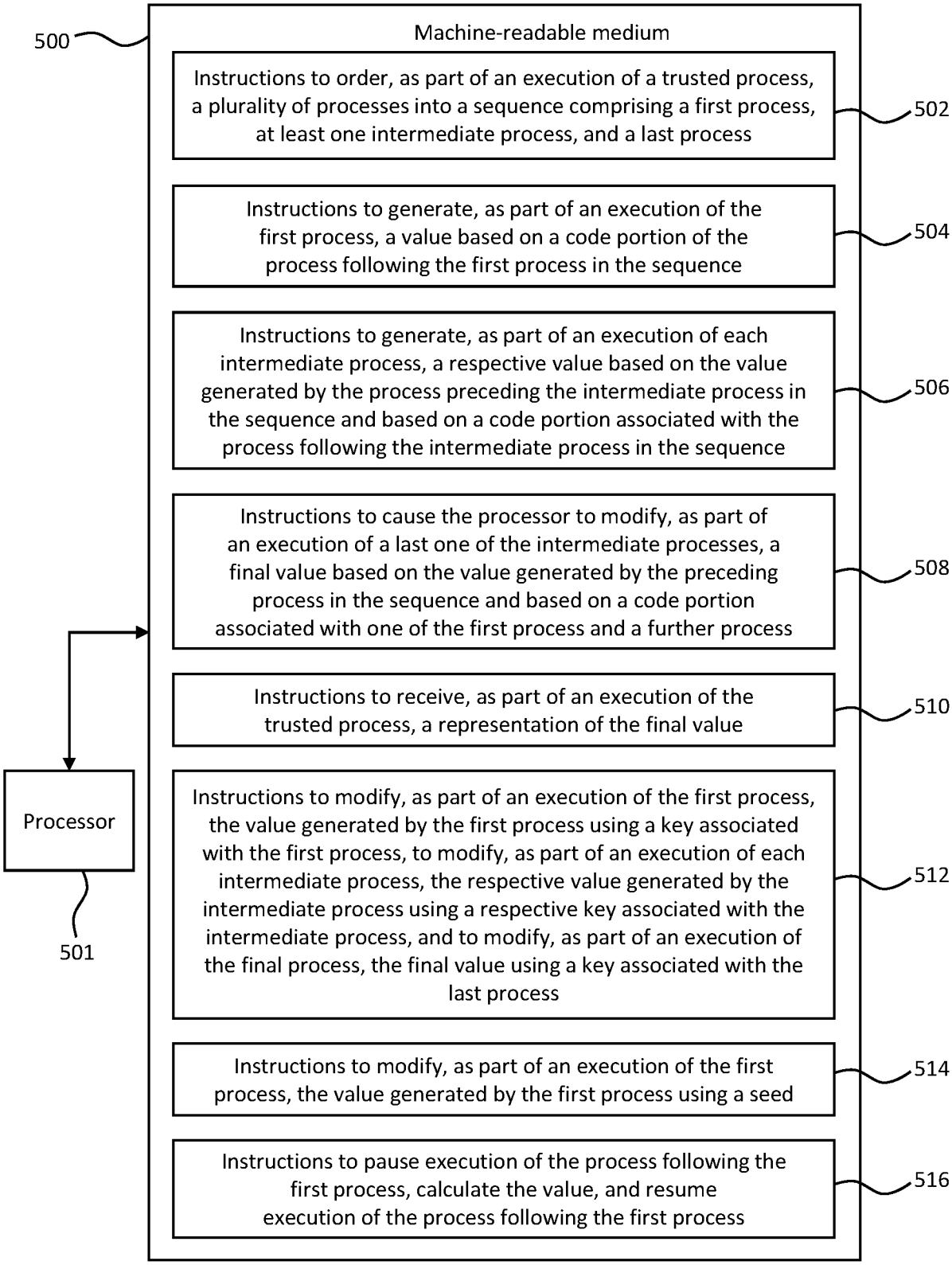
500
Machine-readable medium
Instructions to order, as part of an execution of a trusted process, a plurality of processes into a sequence comprising a first process, at least one intermediate process, and a last process
502
Instructions to generate, as part of an execution of the first process, a value based on a code portion of the process following the first process in the sequence
504
Instructions to generate, as part of an execution of each intermediate process, a respective value based on the value generated by the process preceding the intermediate process in the sequence and based on a code portion associated with the process following the intermediate process in the sequence
506
Instructions to cause the processor to modify, as part of an execution of a last one of the intermediate processes, a final value based on the value generated by the preceding process in the sequence and based on a code portion associated with one of the first process and a further process
508
Instructions to receive, as part of an execution of the trusted process, a representation of the final value
510
Instructions to modify, as part of an execution of the first process, the value generated by the first process using a key associated with the first process, to modify, as part of an execution of each intermediate process, the respective value generated by the intermediate process using a respective key associated with the intermediate process, and to modify, as part of an execution of the final process, the final value using a key associated with the last process
512
Instructions to modify, as part of an execution of the first process, the value generated by the first process using a seed
514
Instructions to pause execution of the process following the first process, calculate the value, and resume execution of the process following the first process
516
Processor
501

FIG. 5

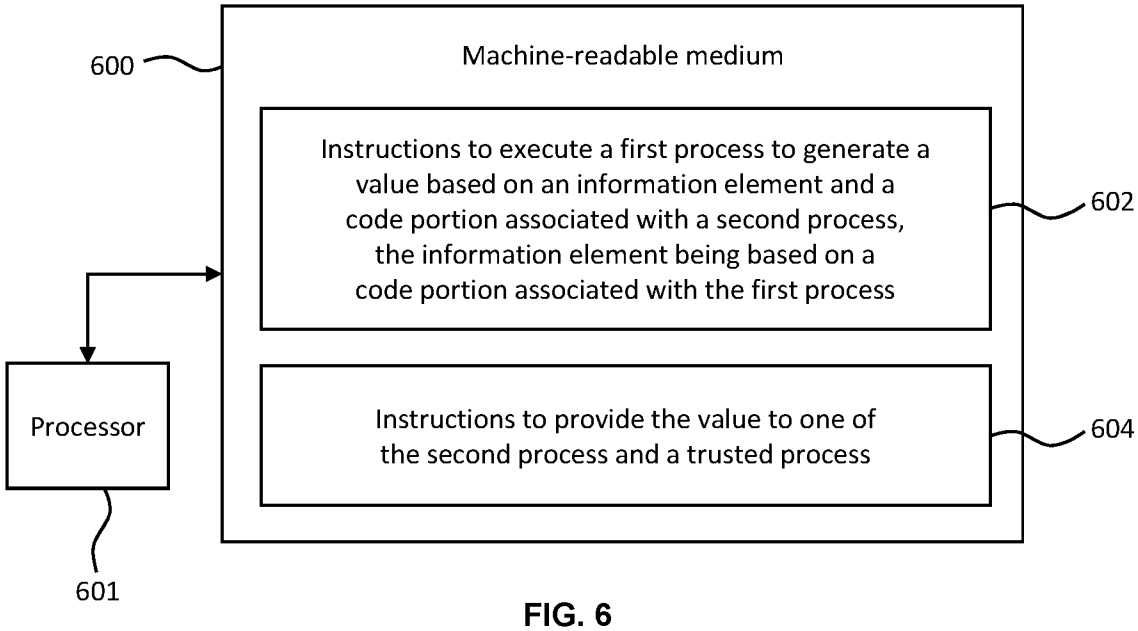
600
Machine-readable medium
Instructions to execute a first process to generate a value based on an information element and a code portion associated with a second process, the information element being based on a code portion associated with the first process
602
Instructions to provide the value to one of the second process and a trusted process
604
Processor
601

FIG. 6

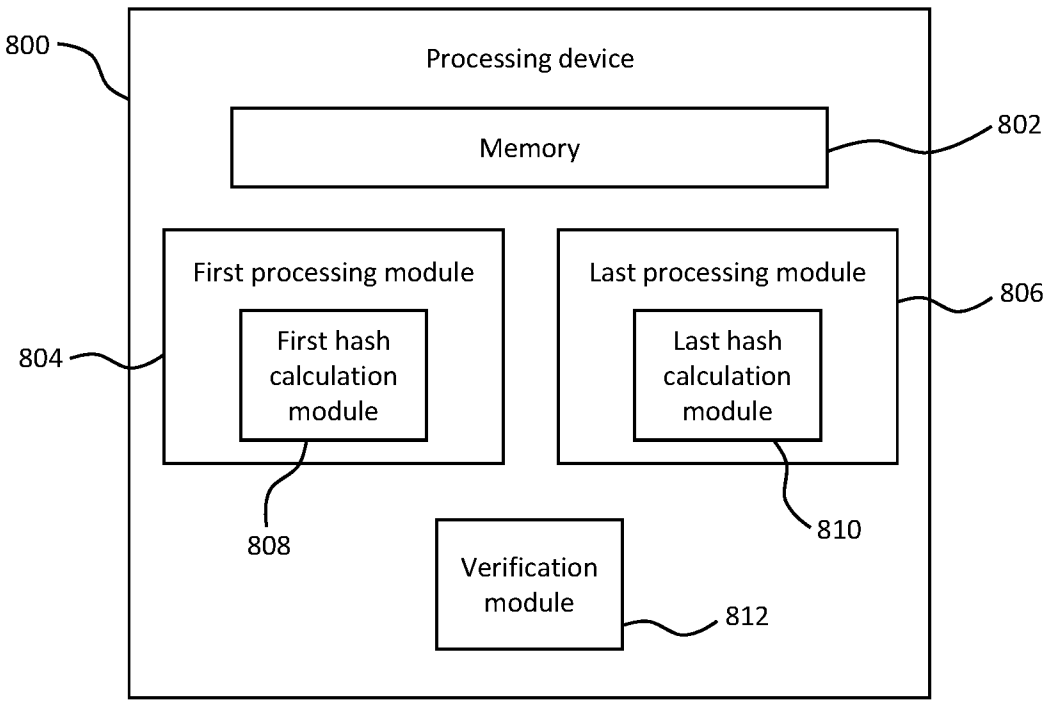
800
Processing device
Memory
802
First processing module
First hash calculation module
804
Last processing module
Last hash calculation module
806
808
810
Verification module
812

FIG. 8

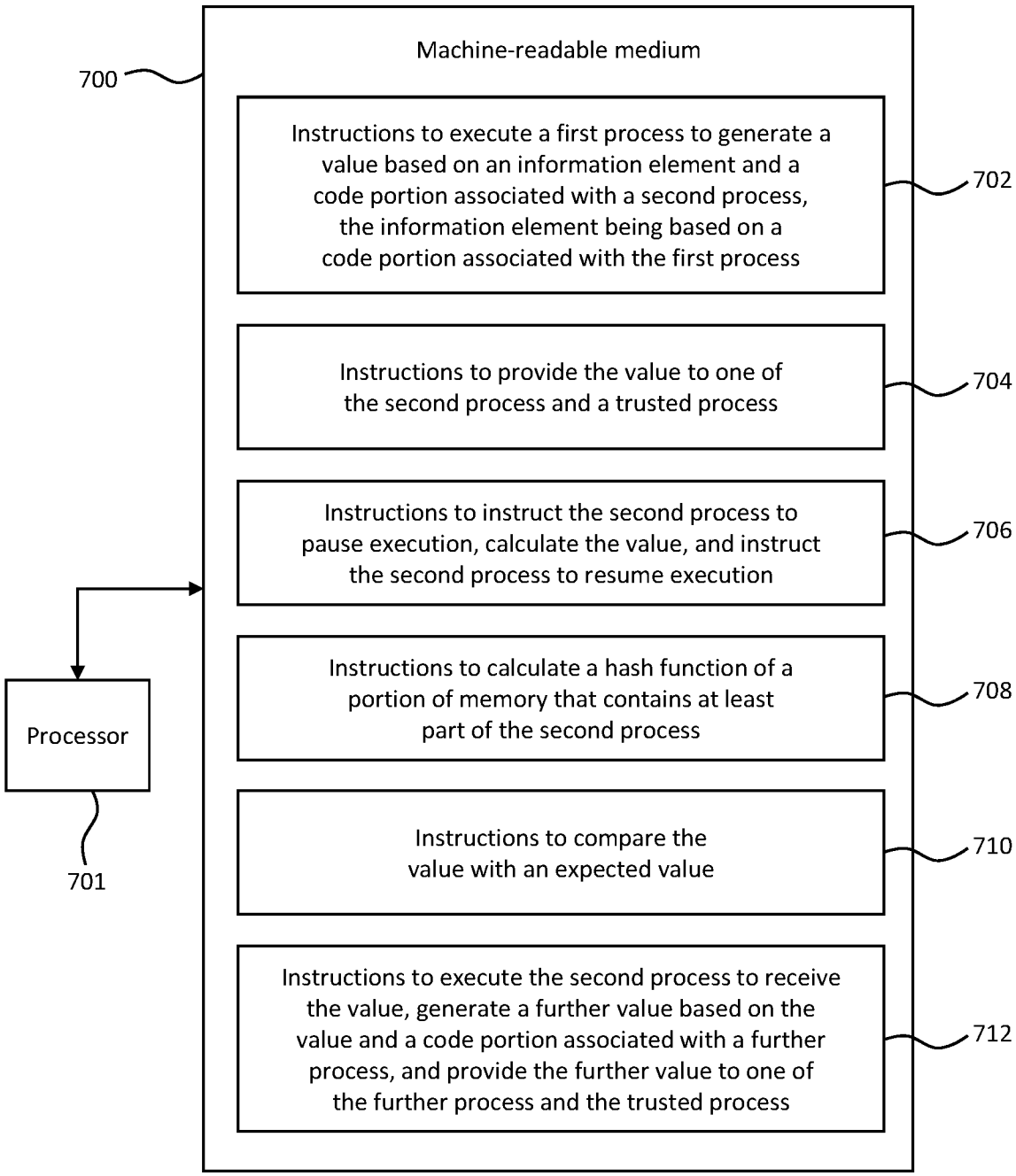
Machine-readable medium
700
Instructions to execute a first process to generate a value based on an information element and a code portion associated with a second process, the information element being based on a code portion associated with the first process
702
Instructions to provide the value to one of the second process and a trusted process
704
Instructions to instruct the second process to pause execution, calculate the value, and instruct the second process to resume execution
706
Instructions to calculate a hash function of a portion of memory that contains at least part of the second process
708
Instructions to compare the value with an expected value
710
Instructions to execute the second process to receive the value, generate a further value based on the value and a code portion associated with a further process, and provide the further value to one of the further process and the trusted process
712
Processor
701

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 2017/028156

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/06 *(2006.01)*
G06F 21/57 *(2013.01)*

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 9/00, 9/06, 21/00, 21/30, 21/44, 21/50, 21/57, 21/60-21/64

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

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

PatSearch, esp@cenet, USPTO, Google

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2010/0318639 A1 (PAUL JAUQUET et al.) 16.12.2010	1-15
A	US 2016/0191246 A1 (INTEL CORPORATION) 30.06.2016	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 document but published on or after the international filing date

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 December 2017 (15.12.2017)	28 December 2017 (28.12.2017)

Name and mailing address of the ISA/RU:	Authorized officer
Federal Institute of Industrial Property, Berezhkovskaya nab., 30-1, Moscow, G-59, GSP-3, Russia, 125993 Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37	T. Kiseleva Telephone No. 8 (495)-531-64-81