SCHEDULING METHOD AND APPARATUS FOR A PLURALITY OF SECURITY SERVICES, DEVICE, AND MEDIUM

TECHNICAL FIELD

[0001] The present disclosure relates to the field of information security and trusted computing, and in particular, to a scheduling method and apparatus for a plurality of security services, and a device.

BACKGROUND

[0002] With improvement of processor performance of a system on chip and diversification of system forms, application scenarios of a trusted execution environment (TEE) have expanded from a single-core processor and a single security service to a complex environment involving a multi-core processor and virtualization of a rich operating system (or, a rich execution environment, REE). Therefore, it is necessary to simultaneously run a plurality of mutually isolated security services (or, secure partitions, SPs) in a secure world.

[0003] In related technologies, to implement a plurality of SPs on a TEE side, mutual isolation between the plurality of SPs is implemented by performing virtualization extension at an exception level (e.g., exception level 2, EL2). However, the foregoing manner is only applicable to a high-end system on chip SoC with EL2 support, and a plurality of SPs cannot be deployed on a low-end SoC, thus the low-end SoC cannot implement scheduling of a plurality of SPs.

SUMMARY

[0004] To resolve the foregoing technical problem, the present disclosure provides a scheduling method for a plurality of security services, so as to resolve a problem that a low-end SoC cannot implement scheduling of a plurality of SPs.

[0005] According to a first aspect of the present disclosure, a scheduling method for a plurality of security services is provided, including:

[0006] generating a first secure monitor call instruction based on a target task being executed

by a normal world;

[0007] in response to the first secure monitor call instruction, switching from a first current exception level of the normal world to a highest exception level;

[0008] determining, based on a target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs;

[0009] determining, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world, where the target security service is used for processing the target task; and

[0010] switching from the highest exception level to a target exception level of the secure world corresponding to the target security service, and starting the target security service at the target exception level of the secure world to execute the target task.

[0011] According to a second aspect of the present disclosure, a scheduling apparatus for a plurality of security services is provided, including:

[0012] a first generation module, configured to generate a first secure monitor call instruction based on a target task being executed by a normal world;

[0013] a first processing module, configured to, in response to the first secure monitor call instruction, switch from a first current exception level of the normal world to a highest exception level;

[0014] a first determining module, configured to determine, based on a target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs;

[0015] a second determining module, configured to determine, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world, where the target security service is used for processing the target task; and

[0016] a second processing module, configured to switch from the highest exception level to a target exception level of the secure world corresponding to the target security service, and start the target security service at the target exception level of the secure world to execute the target task.

[0017] According to a third aspect of the present disclosure, a computer readable storage

medium is proposed. The storage medium stores a computer program, and the computer program is used for implementing the scheduling method for a plurality of security services according to the first aspect described above.

[0018] According to a fourth aspect of the present disclosure, an electronic device is proposed. The electronic device includes: a processor; and a memory, configured to store instructions executable by the processor. The processor is configured to read the executable instructions from the memory, and execute the instructions to implement the scheduling method for a plurality of security services according to the first aspect described above.

[0019] According to a fifth aspect of the present disclosure, a computer program product is proposed. When instructions in the computer program product are executed by a processor, the scheduling method for a plurality of security services according to embodiments in the first aspect of the present disclosure is implemented.

[0020] According to the technical solutions provided in the embodiments of the present disclosure, a secure monitor call instruction may be generated based on a task being executed in the normal world; in response to the secure monitor call instruction, switching may be performed from a current exception level to an EL3 level; the target processor group to which a processor belongs may be determined at the EL3 level based on the processor that initiates the secure monitor call instruction; further, the target security service corresponding to the target processor group may be determined from the plurality of security services included in the secure world; and then switching may be performed from EL3 to an exception level (for example, S-EL0/S-EL1) of the target security service, and the target security service may be started at that exception level to execute the task. In other words, since according to this scheme, a plurality of SPs can be deployed in the secure world and scheduling of the target SP at the EL3 level can be implemented, this scheme does not need to rely on virtualization extension, and a plurality of SPs can be deployed on a low-end SoC by adopting this scheme, so as to implement isolated scheduling of the plurality of SPs, thereby broadening the application scope of the plurality of SPs.

BRIEF DESCRIPTION OF THE DRAWINGS

[0021] FIG. 1 is a schematic diagram of an architecture of a scheduling system for a plurality of security services according to an exemplary embodiment of the present disclosure;

[0022] FIG. 2 is a schematic flowchart of a scheduling method for a plurality of security services according to an exemplary embodiment of the present disclosure;

[0023] FIG. 3 is a schematic flowchart of a scheduling method for a plurality of security services according to another exemplary embodiment of the present disclosure;

[0024] FIG. 4 is a schematic flowchart of a scheduling method for a plurality of security services according to still another exemplary embodiment of the present disclosure;

[0025] FIG. 5 is a schematic flowchart of a scheduling method for a plurality of security services according to yet another exemplary embodiment of the present disclosure;

[0026] FIG. 6 is a schematic timing diagram of a scheduling method for a plurality of security services according to another exemplary embodiment of the present disclosure;

[0027] FIG. 7 is a schematic diagram of a structure of a scheduling apparatus for a plurality of security services according to an exemplary embodiment of the present disclosure; and

[0028] FIG. 8 is a diagram of a structure of an electronic device according to an exemplary embodiment of the present disclosure.

DETAILED DESCRIPTION OF THE EMBODIMENTS

[0029] To explain the present disclosure, exemplary embodiments of the present disclosure are described below in detail with reference to accompanying drawings. Obviously, the embodiments described are merely some, rather than all of embodiments of the present disclosure. It should be understood that the present disclosure is not limited by the exemplary embodiments.

[0030] It should be noted that, unless otherwise specifically stated, the relative arrangement of components and steps, numerical expressions, and numerical values set forth in these embodiments do not limit the scope of the present disclosure.

[0031] Application overview

[0032] A vehicle assisted driving system generally adopts an ARM TrustZone architecture, which divides hardware resources into two worlds through secure isolation, which respectively are a secure world and a non-secure world, where the non-secure world may also be referred to as a normal world.

[0033] The secure world and the non-secure world operate completely independently, and the two worlds can share hardware resources such as processors, memory, and peripherals. A secure

operating system is run in the secure world to process sensitive data or execute key security tasks, such as key management, biometric authentication, or payment protection. Moreover, the normal world runs a normal operating system (that is, a rich operating system, such as Linux or Android) to execute routine tasks. Access authority of the non-secure world is strictly restricted, and resources of the secure world cannot be directly accessed, thereby effectively preventing malicious software or attackers from stealing sensitive information.

[0034] In the ARM TrustZone architecture, a processor implements isolation between different privilege levels through exception levels (ELs). In the ARM TrustZone architecture, the processor supports a plurality of exception levels (ELs), generally including EL0 (a user mode), EL1 (an operating system mode or a kernel mode), EL2 (a virtualization mode), and EL3 (a secure monitor mode). EL3, serving as the highest exception level, is responsible for managing switching between the secure world and the normal world, and for scheduling secure resources. A larger number corresponding to the exception level indicates higher privilege authority of software. Different exception levels may run in the secure world or the normal world, specifically depending on a current world state of the processor.

[0035] EL0 is the lowest privilege level, and is used for running a normal application program. EL0 cannot directly access hardware resources, and needs to indirectly apply for the hardware resources through a system call interface of EL1.

[0036] EL1 is the second-lowest privilege level, and is used for running an operating system kernel (such as a Linux kernel or an Android kernel) and device drivers. EL1 may run in a secure state and a non-secure state, and thus includes two states, which respectively are Secure EL1 (that is, S-EL1) and non-secure EL1 (that is, NS-EL1), where S-EL1 is EL1 in the secure world, and NS-EL1 is EL1 in the non-secure world. S-EL1 is used for running the kernel of the secure operating system (such as a trusted execution environment (e.g., open portable trusted execution environment, OP-TEE)), so as to manage sensitive tasks (such as cryptographic computation or identity tasks) of the secure world. NS-EL1 is used for running a kernel of the normal operating system (such as Android or Linux), so as to manage resource requests of the normal application program.

[0037] EL2 is a medium privilege level, and is used for running a virtualization monitoring program (Hypervisor), so as to implement virtualized deployment of a plurality of operating

systems. EL2 may manage operating environments of EL1 and EL0. A Hypervisor running in a non-secure EL2 cannot access resources of the secure world, but can communicate with EL3 of the secure world through a secure monitor call (SMC) instruction, so as to implement security verification in a virtualization scenario.

[0038] EL3 (a monitor mode) is the highest privilege level, is a core control layer of the ARM TrustZone architecture, and only runs in the secure world to be responsible for switching between the non-secure world and the secure world and for management and control of the security policy. EL3 has the highest hardware access authority, and is used for running a secure monitor. The secure monitor running in EL3 serves as a gateway between the non-secure world and the secure world, and can perform switching between the secure world and the non-secure world.

[0039] In the ARM TrustZone architecture, when a plurality of security services need to be supported, a conventional scheme relies on virtualization extension at EL2 level to implement isolation between the security services. This scheme requires that hardware needs to have EL2 support, that is, a high-end system on chip (SoC) is required for deployment. However, a large quantity of low-end SoCs only support three exception levels, that is, EL0, EL1 and EL3, and lack a virtualization capability at the EL2 level. As a result, isolation and scheduling of a plurality of security services cannot be implemented, so that application scenarios of a plurality of secure operating systems cannot be satisfied.

[0040] Exemplary system

[0041] FIG. 1 is a schematic diagram of an architecture of a scheduling system for a plurality of security services according to an exemplary embodiment of the present disclosure. As shown in FIG. 1, the system includes a plurality of normal operating systems and a plurality of processor groups of a normal world, and a plurality of security services (SPs) of a secure world. Each normal operating system corresponds to a different processor group among the plurality of processor groups, and each processor group corresponds to a different security service among the plurality of security services.

[0042] For example, the normal operating system may be Linux or Android, and may run at an NS-EL1 level of the normal world. This system is used for processing routine tasks of a user. When a normal operating system on a certain processor needs to trigger a secure task during operation, a secure monitor may be notified through a secure monitor call instruction.

[0043] A target processor among the plurality of processor groups is configured to: generate a first secure monitor call instruction based on a target task being executed by the normal world; in response to the first secure monitor call instruction, switch from a first current exception level of the normal world to a highest exception level; determine, based on the target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs; determine, based on the target processor group, a target security service corresponding to the target processor group from the plurality of security services included in the secure world, where the target security service is used for processing the target task; and switch from the highest exception level to a target exception level of the secure world corresponding to the target security service, and start the target security service at the target exception level of the secure world to execute the target task. In some embodiments, the target processor group may perform the target task in one of the plurality of normal operating systems. In some embodiments, the plurality of processor groups may run a secure monitor at the highest exception level. The secure monitor may be an interface between the secure world and the normal world. In some embodiments, the secure monitor includes a security service scheduling module which, in response to the first secure monitor call instruction, determine, based on the target processor that initiates the first secure monitor call instruction, the target processor group to which the target processor belongs. In some embodiments, the first current exception level and the target exception level may be two different states of the same privilege level.

[0044] For example, as shown in FIG. 1, taking the normal world including an operating system 110 and an operating system 111 as an example, the secure world includes SP#0 and SP#1, and the processor groups include a processor group 0 and a processor group 1. The operating system 110 corresponds to the processor group 0, and the processor group 0 corresponds to the security service SP#0 of the secure world. The operating system 111 corresponds to the processor group 1, and the processor group 1 corresponds to the security service SP#1 of the secure world. It should be noted that this embodiment of the present disclosure does not limit quantities of the normal operating systems, the security services, and the processor groups, and exemplary description is made in this embodiment of the present disclosure by using an example in which the quantity of each is only two.

[0045] For example, when a certain processor in the processor group 0 is executing a security-

related task on the operating system 110, such as a sensitive task of key generation, digital signature verification, biometric comparison, or secure storage access, this processor may generate the first secure monitor call instruction to request switching to the secure world to process the sensitive task. This processor traps from the NS-EL1 level, at which the operating system 110 is located, to an EL3 level, so that a security service scheduling module on a secure monitor layer (that is, a secure monitor) running at the EL3 level determines, based on the processor that initiates the request, the processor group 0 to which this processor belongs, so as to match to the security service SP#0 pre-bound to the processor group 0. This processor switches from the EL3 level to a target exception level corresponding to SP#0, such as an S-EL1 level, and starts SP#0 to process the current sensitive task.

[0046] When a processor in the processor group 1 initiates a security service call, the security service SP#1 may be correspondingly matched according to the foregoing flow to complete processing of the target task, so as to implement isolated scheduling of secure tasks of different processor groups, thereby avoiding interference between different security services. Meanwhile, there is no need for a single security service to sequentially process secure tasks of all processor groups, thereby improving processing efficiency for the secure tasks.

[0047] After task processing is completed, it may switch back to the corresponding exception level of the normal world according to the same routing rule, to return a processing result to the normal operating system. Thus, according to this scheme, isolated operation and independent isolation of a plurality of security services can be implemented without relying on EL2 virtualization extension, thereby satisfying deployment requirements of a plurality of security services on a low-end SoC hardware platform.

[0048] It should be noted that, for detailed description of the foregoing system executing a scheduling method for a plurality of security services provided in an embodiment of the present disclosure, reference may be made to detailed description in the method embodiment section below, and details are not described in this embodiment of the present disclosure.

[0049] Exemplary method

[0050] FIG. 2 is a schematic flowchart of a scheduling method for a plurality of security services according to an exemplary embodiment of the present disclosure. This embodiment may be applied to an electronic device or a chip of the electronic device.

[0051] For example, as shown in FIG. 2, the foregoing method may include the following steps.

[0052] Step 201: Generating a first secure monitor call instruction based on a target task being executed by a normal world.

[0053] In some embodiments, the target task refers to a security-related task that needs to be processed by the secure world, such as a sensitive operation of key generation, digital signature verification, biometric comparison, or secure storage access. When a processor of the normal world (that is, the non-secure world) is executing the target task, if an application program or an operating system kernel of the normal world detects a security requirement, a security operation request may be initiated to the kernel of the normal world through a system call interface. Based on the security operation request, the processor generates and triggers the first secure monitor call (SMC) instruction to trigger the processor to switch from the normal world to the secure world. The first secure monitor call instruction may carry task information such as a task identifier, a task type, and an input parameter pointer, so that the secure world can identify specific content of the task to be processed.

[0054] Step 202: In response to the first secure monitor call instruction, switching from a first current exception level of the normal world to the highest exception level.

[0055] In some examples, after the first secure monitor call instruction is triggered by the processor of the normal world, in response to the first secure monitor call instruction, the processor switches from the normal world to a secure monitor layer (or a secure monitor), such as a core boot program (e.g., BootLoader stage 3, BL31) in an ARM architecture. In other words, the processor traps from the first current exception level (typically NS-EL1 or NS-EL0) to the highest exception level (EL3). During an exception level switching process of the processor, the processor automatically saves current execution context (for example, including a general-purpose register, a program counter, and a program status register) of the normal world to a stack at the first current exception level, and loads execution context at the EL3 level after switching to the EL3 level.

[0056] Step 203: Determining, based on a target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs.

[0057] In some embodiments, for a chip system including a plurality of processors, during a system startup phase, the plurality of processors may be pre-divided into a plurality of processor groups, and grouping information may be stored in a system configuration table, with each

processor group having a unique processor group identifier. The target processor described above is a physical processor core that executes the target task in the normal world.

[0058] In some examples, after the target processor switches from the normal world to the EL3 level, execution authority is transferred to the secure monitor layer. A physical identifier (such as a processor ID) of the target processor is obtained by calling a security service scheduling module (secure partition manager core, SPMC) in the secure monitor layer, and the processor group identifier of the target processor group to which the target processor belongs is determined by querying the system configuration table based on the physical identifier, or is obtained by directly reading a hardware register (such as an MPIDR_EL1 register) of the target processor. The processor group identifier may also be referred to as a CPU cluster ID.

[0059] Step 204: Determining, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world.

[0060] The target security service is used for processing the target task.

[0061] In some embodiments, when initialization is performed by the secure monitor layer during the system startup phase, each processor group may be pre-bound to at least one security service, so as to establish a binding relationship between processor groups and security services. In this way, physical isolation and refined scheduling of a plurality of security services can be implemented, thereby avoiding cross-group resource contention and unauthorized access.

[0062] For example, a security service identifier associated with a target processor group identifier is obtained based on the target processor group and the preset binding relationship between processor groups and security services, and further the target security service is determined based on the security service identifier. The binding relationship is used for indicating a correspondence between each processor group identifier and a security service identifier bound thereto, and the security service identifier is used for uniquely identifying a respective security service instance in the secure world.

[0063] Step 205: Switching from the highest exception level to a target exception level of the secure world corresponding to the target security service, and starting the target security service at the target exception level of the secure world to execute the target task.

[0064] In some embodiments, the target security service runs at a specific exception level of the

secure world, typically S-EL1 or S-EL0, that is, the target exception level in a secure state. After the target security service (that is, a target SP) is determined, the secure monitor layer determines, based on configuration information of the target security service, a target exception level at which the target security service runs, and triggers the target processor to perform an exception level switching operation from EL3 to the target exception level. During the exception level switching process of the target processor, the target processor first saves execution context of the EL3 level to a stack at the EL3 level; and then, after the target processor traps to the target exception level of the secure state, loads or restores the execution context of the target security service. Specifically, the secure monitor layer obtains context information, such as an entry address, a stack pointer configuration, memory mapping information, and a permission configuration, from a secure partition descriptor of the target security service, and loads the information into a corresponding register of the target processor, so as to complete context switching. Subsequently, the target processor jumps to an entry point of the target security service, and starts the target security service to execute the target task.

[0065] It may be understood that the target security service is deployed in the secure world in the form of a secure partition, and each secure partition has an independent address space and resource view. The SPMC in the secure monitor layer is responsible for lifecycle management and scheduling of the secure partitions. When starting the target security service, the SPMC activates corresponding secure partition context, so that the secure partition obtains execution authority of the processor.

[0066] For example, exemplary description is made by taking a cooperative scenario of an in-vehicle infotainment system and a vehicle control system as an example. In this cooperative scenario, the electronic device adopts a low-end automotive-grade SoC that only supports EL0, EL1, and EL3. The chip system is configured with two processor groups: a processor group A includes two processor cores, runs an Android operating system, and is responsible for infotainment functions; and a processor group B includes one processor core, runs a real-time operating system, and is responsible for processing vehicle control instructions. There are two security services deployed in the secure world: security service 1 (bound to the processor group A), being used for managing decryption of digital-rights-protected content and payment authentication; and security service 2 (bound to the processor group B), being used for managing

signature verification and key storage of the vehicle control instructions.

[0067] When a user initiates a video-on-demand request in the normal world, an Android application program detects that requested content is encrypted, and initiates a security operation request to an NS-EL1 kernel through a call interface, so as to generate an SMC instruction to trigger world switching. After the processor traps from NS-EL1 to EL3, the secure monitor determines, based on an ID of a processor core that initiates the SMC instruction, that the processor core belongs to the processor group A, and queries a mapping table to learn that the target security service is security service 1. Further, the processor switches from EL3 to security service 1 at the S-EL1 level, to execute a content decryption task. Meanwhile, a vehicle control unit may independently call security service 2 through the processor group B to perform control instruction authentication, and execution of the two security services is mutually isolated and does not interfere with each other. In this way, according to a deployment scheme of the present disclosure, concurrent scheduling of a plurality of security services on a hardware platform lacking EL2 support is achieved, thereby satisfying requirements for functional safety and information security of an in-vehicle system. Meanwhile, for a high-end system on chip SoC with EL2 support, without relying on the virtualization extension of EL2, concurrent scheduling of a plurality of mutually isolated security services can still be independently achieved by adopting the scheme provided in the present disclosure.

[0068] According to the technical solutions provided in the embodiments of the present disclosure, when the normal world needs to call a security service, the processor switches from a current exception level to the EL3 level. By fully utilizing a scheduling capability at the highest exception level EL3, the processor group to which the processor belongs is identified and the target security service corresponding to the processor group is determined. Further, the processor directly switches to an exception level at which the security service is located, to execute the target task. According to this scheme, without relying on EL2 virtualization extension, physical isolation and independent scheduling of a plurality of security services are implemented by binding the processor group with different security services and by uniformly implementing security scheduling at the EL3 level, which avoids mutual interference during operation of different security services, and can also support concurrent processing of a plurality of security services, thereby effectively improving overall processing efficiency of a secure task. In this way, this

scheme is compatible with a low-end hardware platform not supporting the EL2 virtualization extension, satisfying deployment requirements of isolation of a plurality of security services on a low-end SoC platform; and can also provide an additional multiple-secure-service deployment option on a high-end platform, accommodating different hardware scenario requirements. Therefore, this scheme can satisfy requirements of the in-vehicle system for functional safety and information security.

[0069] In some embodiments, as shown in FIG. 3, on the basis of the embodiments shown in FIG. 2, the foregoing step 204 may specifically include the following steps.

[0070] Step 2041: Obtaining a processor group identifier of the target processor group.

[0071] For example, based on a secure monitor layer running at the EL3 level, the processor group identifier (such as the CPU cluster ID) of the target processor group to which the target processor belongs is obtained by reading the hardware register (such as an MPIDR_EL1 register) of the target processor or querying the system configuration table. The processor group identifier serves as a key index for a subsequent query of a binding relationship mapping table, so as to precisely locate the corresponding target security service. Since the obtaining process of the processor group identifier is completed at the EL3 level, credibility and integrity of identifier information are ensured, thereby preventing malicious tampering from the normal world.

[0072] Step 2042: Determining, by calling a security service scheduling module at the highest exception level, a target service identifier corresponding to the processor group identifier based on a preset context table.

[0073] The target service identifier is used for indicating the target security service, and the context table is used for indicating mapping relationships between different groups of processors and each of the plurality of security services.

[0074] In some examples, the security service scheduling module (the secure partition manager core, SPMC) is a functional component integrated in the secure monitor layer at the EL3 level, and a core function thereof is to be responsible for managing and scheduling a plurality of security services in a system. The security service scheduling module has high-priority access authority to system resources, and can directly interact with a hardware security mechanism and the security services. The security service scheduling module implements centralized management of the mapping relationships between different processor groups and the security services by maintaining

the preset context table. The context table stores the mapping relationships between the processor group identifiers and the security service identifiers. For example, the context table may be implemented by using a data structure such as a hash table or an ordered array, so as to support fast query.

[0075] In some embodiments, after the processor group identifier is obtained by the secure monitor layer running at the EL3 level, the security service scheduling module is called at the EL3 level. The security service scheduling module performs a precise lookup in the context table based on the processor group identifier, so as to determine the target service identifier corresponding to the processor group identifier. The target service identifier uniquely indicates the target security service that needs to be called by the current processor group, thereby laying a foundation for accurate starting and execution of a subsequent security service.

[0076] In some examples, the target service identifier may be an index identifier of the security service. After the determining of the target service identifier is completed, the security service scheduling module may return the target service identifier to an exception handling flow of the secure monitor layer, so as to facilitate subsequent obtaining of complete configuration information of the target security service, ensuring that a subsequent exception level switching and service starting flow can be smoothly performed.

[0077] Based on the foregoing embodiments, since the security service scheduling module is disposed at the highest exception level EL3, and the context table recording the mapping relationships between the processor groups and the security services is maintained, after the processor group identifier is obtained by the secure monitor layer, the security service scheduling module can be called at the EL3 level to precisely look up the context table based on the processor group identifier, to determine the target service identifier, thereby ensuring that the target security service required by the current processor group is accurately started and executed. In this way, not only is efficiency and accuracy of security service scheduling improved, avoiding a security risk that may be caused by a scheduling error or delay, but also reliable underlying support is provided for collaborative work of a plurality of security services in a complex system environment, enhancing overall security and stability of the system.

[0078] In some optional embodiments, as shown in FIG. 4, on the basis of the embodiments shown in FIG. 2, the scheduling method for a plurality of security services provided in the

embodiments of the present disclosure may further include the following steps.

[0079] Step 206: In response to a system on chip including a plurality of processor groups, scheduling and starting, at the highest exception level, images of security services corresponding to all processor groups, and obtaining registration information of the security services.

[0080] The registration information includes at least one of identifiers of the security services and processor group identifiers bound to the security services.

[0081] In some embodiments, the starting of the images of the security services is completed during a system initialization phase. After completing power-on startup, the system on chip first enters an initialization flow of the secure monitor layer at the EL3 level, and the secure monitor layer may traverse all security service entries pre-defined in a device tree file, and sequentially load and start the images of the security services. After the images corresponding to the security services are loaded to a pre-allocated secure memory region, primary cores of the processor groups respectively start the images of the corresponding security services, and automatically read, during starting of the images, identifiers of the security services (such as UUIDs and id fields of the security services) stored in a device tree, and read hardware registers corresponding to the primary cores to obtain the processor group identifiers of the processor groups. Moreover, basic information and initial context information of the respective security services may also be obtained. The basic information at least includes attributes, names, versions, and the like. The initial context information may include operating states and operating locations of the security services. Further, the various information obtained in the foregoing process may be used as registration information, so as to register the security services.

[0082] Step 207: Generating registration requests based on the registration information of the security services, and initiating, by the security services, respective registration requests to the security service scheduling module.

[0083] The registration request carries the registration information of the security service.

[0084] In some embodiments, security verification may be performed on the registration information of the security services, and after the verification is completed, the registration request may be generated based on the registration information, and then the registration request may be sent to the security service scheduling module through a communication channel pre-established between the security service and the security service scheduling module.

[0085] Step 208: In response to the registration requests, generating and saving the context table by the security service scheduling module based on the registration information of the security services.

[0086] The context table is used for indicating the mapping relationships between different groups of processors and each of the plurality of security services.

[0087] In some examples, the security service scheduling module initializes a global array, and writes the registration information of all security services (including the identifiers of the security services, the processor group identifiers bound to the security services, the basic configuration, and the like) one by one into corresponding positions of the array; and generates corresponding index identifiers for the security services, so as to obtain the context table recording the mapping relationships between different processor groups and the security services. The context table is saved in a specific storage region at the EL3 level, and is managed and maintained by the security service scheduling module.

[0088] Based on the foregoing embodiments, batch registration of a plurality of security services and generation of the context table are performed by the security service scheduling module running at the EL3 level, thereby implementing unified centralized management of the mapping relationships between the processor groups and the security services. Thus, when a certain security service needs to be called during system operation, the context table can be quickly queried based on the processor group identifier, to obtain the accurate security service identifier.

[0089] In some optional embodiments, as shown in FIG. 5, on the basis of the embodiments shown in FIG. 2, the scheduling method for a plurality of security services provided in the embodiments of the present disclosure may further include the following steps.

[0090] Step 209: Generating a second secure monitor call instruction in response to completion of execution of the target task.

[0091] In some embodiments, after the target security service completes processing of the target task at the target exception level of the secure world, an execution result is generated and is prepared to be returned to the normal world. In this case, the target security service running at the secure exception level actively initiates a world switching request by triggering the second secure monitor call instruction (SMC), so as to hand over execution authority to the secure monitor layer. To be specific, the second secure monitor call instruction is actively triggered by the secure world

side, used for indicating that task execution is completed and it is requested to return to the normal world. A function of the second secure monitor call instruction corresponds to that of the first secure monitor call instruction initiated by the normal world, and the second secure monitor call instruction and the first secure monitor call instruction together constitute a complete bidirectional interaction mechanism between the normal world and the secure world.

[0092] For example, after the target security service completes the processing of the target task at the target exception level of the secure world, the target security service writes information such as a task execution result and a return status into a preset register, and generates the second secure monitor call instruction, to request switching back to the highest exception level and return the result to the normal world.

[0093] Step 210: In response to the second secure monitor call instruction, switching from a second current exception level of the secure world to the highest exception level, and parsing the second secure monitor call instruction at the highest exception level to obtain a task execution result of the target task.

[0094] In some examples, after the second secure monitor call instruction is triggered by the target security service in the secure world, in response to the second secure monitor call instruction, the processor traps from the second current exception level (such as S-EL1/S-EL0) of the secure world to the EL3 level, and parses the second secure monitor call instruction at the EL3 level, so as to identify the second secure monitor call instruction as a return request after completion of execution of the task in the secure world; and after the call type is determined, reads the task execution result of the target task corresponding to the return request from a preset register.

[0095] In some embodiments, the foregoing step 210 may specifically include: parsing the second secure monitor call instruction by calling a security service scheduling module at the highest exception level, to obtain the task execution result of the target task.

[0096] In some examples, during parsing of the second secure monitor call instruction at the EL3 level, the security service scheduling module (SPMC) in the secure monitor layer may be called; the second secure monitor call instruction may be parsed by the SPMC, so as to be identified as a return request after completion of execution of the task in the secure world; and the task execution result of the target task may be obtained from a preset general-purpose register.

[0097] It should be noted that the SPMC may further save current execution context of the target

security service, so that a subsequent task may continue to be executed.

[0098] Step 211: Switching from the highest exception level back to the first current exception level of the normal world, and returning the task execution result to the normal world.

[0099] After the task execution result of the target task is obtained by the secure monitor layer at the EL3 level, the processor switches from the highest exception level EL3 back to the first current exception level (NS-EL1 or NS-EL0) of the normal world, and returns the task execution result to the normal world.

[00100] For example, the task execution result may be rewritten into the preset general-purpose register of the processor by the security service scheduling module (SPMC), and the execution context saved by the normal world before entering EL3 is restored by the SPMC, including a general-purpose register, a program counter, a program status register, and a stack pointer. After completing the context restoration, the processor may switch from the EL3 level back to the first current exception level (for example, NS-EL1 or NS-EL0) of the normal world. After execution is resumed in the normal world, the task execution result of the current security service call may be read from the preset general-purpose register, and then running of subsequent business logic may be continued according to a normal flow, thus completing the entire security service call.

[00101] It should be noted that the foregoing entire switching-and-return process may be fully controlled by the security service scheduling module running at the EL3 level for saving and restoration of context throughout the process. This not only ensures normativeness of the interaction flow, but also avoids leakage of context information between different worlds, thereby further improving security of the system.

[00102] Based on the foregoing embodiments, a return flow actively initiated by the secure world is implemented through the second secure monitor call instruction, and a bidirectional interaction closed loop between the normal world and the secure world is constructed in cooperation with a call flow initiated by the normal world, ensuring that each security service call can accurately complete return of the result, thereby further improving reliability of scheduling of a plurality of security services.

[00103] FIG. 6 is a schematic timing diagram of a scheduling method for a plurality of security services according to another exemplary embodiment of the present disclosure. The timing diagram shows a complete flow, based on an FF-A architecture, in which a secure monitor layer

running at an EL3 level schedules interaction between a normal world and a secure world (SP). This embodiment may be applied to an electronic device, which includes a normal execution environment and a secure execution environment. The normal execution environment corresponds to the normal world, and the secure execution environment corresponds to the secure world. Exemplary description is made by using the secure world including SP#0 and SP#1. The scheduling method for a plurality of security services is introduced below with reference to FIG. 6. As shown in FIG. 6, the method includes the following steps.

[00104] Step 1: The normal world initiates a security service call request.

[00105] When an operating system or an application running in the normal world needs to call a target SP#1, a secure monitor call instruction may be initiated based on the FF-A call protocol, so as to trigger a processor to trap from an NS-EL1 level of the normal world to an EL3 level.

[00106] Step 2: EL3-SPMD receives and forwards the call request.

[00107] After the secure monitor call instruction is received by the secure monitor layer, a secure monitor distribution module SPMD running in the secure monitor layer at the EL3 level (that is, EL3-SPMD) is called to forward the secure monitor call instruction to a security service scheduling module SPMC in the secure monitor layer (that is, EL3-SPMC).

[00108] Step 3: EL3-SPMD receives and forwards the call to EL3-SPMC.

[00109] The secure monitor distribution module running at the EL3 level (EL3-SPMD) forwards the received secure monitor call instruction to EL3-SPMC, which is responsible for subsequent security service scheduling and context management.

[00110] Step 4: EL3-SPMC determines a target SP based on the processor that initiates the secure monitor call instruction.

[00111] EL3-SPMC identifies, based on processor context when the secure monitor call instruction is triggered, the physical processor that initiates the call, and queries a preset mapping relationship based on a hardware identifier (such as a value of an MPIDR register) of the processor, so as to determine a processor group to which the processor belongs; and directly matches, based on a preset binding relationship between processor groups and security services, a target security service (such as SP#1) uniquely bound to the processor group.

[00112] Step 5: EL3-SPMC performs context switching and schedules the target SP for execution.

[00113] Saving context: EL3-SPMC first saves current EL3 context of the current processor of

the normal world.

[00114] Loading context: EL3-SPMC loads execution context (including a stack pointer, a register state, memory mapping information, and the like) of the target security service (SP#1), so that the processor switches from the EL3 level to a secure-world exception level (such as S-EL1) at which the target SP is located.

[00115] Starting the target SP to execute the task: SP#1 runs in the secure world and processes a sensitive task (such as key generation or signature verification) initiated by the normal world.

[00116] Step 6: The target SP completes execution and triggers a return flow.

[00117] For example, after SP#1 completes task processing, an execution result is generated and a return request is actively initiated through the secure monitor call instruction, so that the processor traps from the secure world exception level to EL3 again.

[00118] Step 7: EL3-SPMC restores the context and returns a result.

[00119] Restoring the context: EL3-SPMC receives the return request of SP#1, saves current context of SP#1 (so as to continue scheduling subsequently), and restores the execution context of the normal world that is saved in step 1.

[00120] Returning a call result: EL3-SPMC transfers the execution result of SP#1 to EL3-SPMD through a preset register, and EL3-SPMD forwards the execution result back to the normal world.

[00121] Step 8: The normal world receives the result and completes the call flow.

[00122] The processor returns from EL3 to the normal-world exception level, and the operating system or application of the normal world reads the execution result from the preset register and resumes normal execution. In this case, this security service call flow ends.

[00123] Exemplary apparatus

[00124] FIG. 7 is a schematic diagram of a structure of a scheduling apparatus for a plurality of security services according to an exemplary embodiment of the present disclosure. The apparatus may be disposed in an electronic device such as a terminal device or a server, or on an object such as a vehicle, to implement the scheduling method for a plurality of security services according to any one of the foregoing embodiments of the present disclosure.

[00125] As shown in FIG. 7, an apparatus 300 may include: a first generation module 301, configured to generate a first secure monitor call instruction based on a target task being executed by a normal world; a first processing module 302, configured to, in response to the first

secure monitor call instruction, switch from a first current exception level of the normal world to the highest exception level; a first determining module 303, configured to determine, based on a target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs; a second determining module 304, configured to determine, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world, where the target security service is used for processing the target task; and a second processing module 305, configured to switch from the highest exception level to a target exception level of the secure world corresponding to the target security service, and start the target security service at the target exception level of the secure world to execute the target task.

[00126] In a possible implementation, the second determining module 304 is specifically configured to: obtain a processor group identifier of the target processor group; and determine, by calling a security service scheduling module at the highest exception level, a target service identifier corresponding to the processor group identifier based on a preset context table, where the target service identifier is used for indicating the target security service, and the context table is used for indicating mapping relationships between different groups of processors and each of the plurality of security services.

[00127] In a possible implementation, the apparatus may further include: a first obtaining module, configured to, in response to a system on chip including a plurality of processor groups, schedule and start, at the highest exception level, images of security services corresponding to all processor groups, and obtain registration information of the security services, where the registration information includes at least one of identifiers of the security services and processor group identifiers bound to the security services; a third processing module, configured to generate registration requests based on the registration information of the security services, and initiate, through the security services, respective registration requests to the security service scheduling module, where the registration request carries the registration information of the security service; and a fourth processing module, configured to, in response to the registration requests, generate and save the context table by the security service scheduling module based on the registration information of the security services, where the context table is used for indicating

the mapping relationships between different groups of processors and each of the plurality of security services.

[00128] In a possible implementation, the apparatus may further include: a second generation module, configured to generate a second secure monitor call instruction in response to completion of execution of the target task; a fifth processing module, configured to, in response to the second secure monitor call instruction, switch from a second current exception level of the secure world to the highest exception level, and parse the second secure monitor call instruction at the highest exception level to obtain a task execution result of the target task; and a sixth processing module, configured to switch from the highest exception level back to the first current exception level of the normal world, and return the task execution result to the normal world.

[00129] In a possible implementation, the fifth processing module is specifically configured to parse the second secure monitor call instruction by calling a security service scheduling module at the highest exception level, to obtain the task execution result of the target task.

[00130] For beneficial technical effects corresponding to the exemplary embodiments of this apparatus, reference may be made to the corresponding beneficial technical effects in the exemplary method section described above, and details are not described herein.

[00131] Exemplary electronic device

[00132] FIG. 8 is a diagram of a structure of an electronic device according to an embodiment of the present disclosure. The electronic device includes at least one processor 111 and a memory 112.

[00133] The processor 111 may be a central processing unit (CPU) or another form of processing unit having a data processing capability and/or an instruction execution capability, and may control other components in the electronic device 11 to perform a desired function.

[00134] The memory 112 may include one or more computer program products, which may include various forms of computer readable storage media, such as a volatile memory and/or a non-volatile memory. The volatile memory may include, for example, a random access memory (RAM) and/or a cache. The non-volatile memory may include, for example, a read-only memory (ROM), a hard disk, or a flash memory. One or more computer program instructions may be stored on the computer readable storage medium. The processor 111 may run the one or more computer program instructions to implement the scheduling method for a plurality of security services

according to various embodiments of the present disclosure that are described above and/or other desired functions.

[00135] In an example, the electronic device 11 may further include an input device 113 and an output device 114. These components are interconnected through a bus system and/or other forms of connection mechanisms (not shown).

[00136] The input device 113 may further include, for example, a keyboard or a mouse.

[00137] The output device 114 may output various information to the outside, and may include, for example, a display, a speaker, a printer, a communication network, and a remote output device connected to the communication network.

[00138] Certainly, for simplicity, FIG. 8 shows only some of the components in the electronic device 11 that are related to the present disclosure, and components such as a bus and an input/output interface are omitted. In addition, according to specific applications, the electronic device 11 may also include any other appropriate components.

[00139] Exemplary computer program product and computer readable storage medium

[00140] In addition to the foregoing method and device, embodiments of the present disclosure may also provide a computer program product including computer program instructions that, when run by a processor, cause the processor to perform the steps, of the scheduling method for a plurality of security services according to the embodiments of the present disclosure, that are described in the "Exemplary method" section described above.

[00141] The computer program product may be program code, written with one or any combination of a plurality of programming languages, that is configured to perform the operations in the embodiments of the present disclosure. The programming languages include an object-oriented programming language such as Java or C++, and further include a conventional procedural programming language such as a "C" language or a similar programming language. The program code may be entirely or partially executed on a user computing device, executed as an independent software package, partially executed on the user computing device and partially executed on a remote computing device, or entirely executed on the remote computing device or a secure server.

[00142] In addition, the embodiments of the present disclosure may further relate to a computer readable storage medium, on which computer program instructions are stored. The computer

program instructions, when run by the processor, cause the processor to perform the steps, of the scheduling method for a plurality of security services according to the embodiments of the present disclosure, that are described in the "Exemplary method" section described above.

[00143] The computer readable storage medium may be one readable medium or any combination of a plurality of readable media. The readable medium may be a readable signal medium or a readable storage medium. The readable storage medium includes, for example but is not limited to electrical, magnetic, optical, electromagnetic, infrared, or semiconductor systems, apparatuses, or devices, or any combination of the above. More specific examples (a non-exhaustive list) of the readable storage medium include: an electrical connection with one or more conducting wires, a portable disk, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or a flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the above.

[00144] Basic principles of the present disclosure are described above in combination with specific embodiments. However, advantages, superiorities, and effects mentioned in the present disclosure are merely examples rather than limitations, and it cannot be considered that these advantages, superiorities, and effects are necessary for the embodiments of the present disclosure. In addition, the specific details disclosed above are merely for examples and for ease of understanding, rather than for limitations. The details described above do not limit the present disclosure to necessarily adopting the foregoing specific details for implementation.

[00145] A person skilled in the art may make various modifications and variations to the present disclosure without departing from the spirit and the scope of the present disclosure. In this way, if these modifications and variations of the present disclosure fall within the scope of the claims and equivalent technologies of the claims of the present disclosure, the present disclosure also intends to include these modifications and variations.

CLAIMS

1. A scheduling method for a plurality of security services, comprising:

generating a first secure monitor call instruction based on a target task being executed by a normal world;

in response to the first secure monitor call instruction, switching from a first current exception level of the normal world to a highest exception level;

determining, based on a target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs;

determining, based on the target processor group, a target security service corresponding to the target processor group from the plurality of security services included in a secure world, wherein the target security service is used for processing the target task; and

switching from the highest exception level to a target exception level of the secure world corresponding to the target security service, and starting the target security service at the target exception level of the secure world to execute the target task.

2. The method according to claim 1, wherein the determining, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world comprises:

obtaining a processor group identifier of the target processor group; and

determining, by calling a security service scheduling module at the highest exception level, a target service identifier corresponding to the processor group identifier based on a preset context table, wherein the target service identifier is used for indicating the target security service, and the context table is used for indicating mapping relationships between different groups of processors and each of the plurality of security services.

3. The method according to claim 1 or 2, wherein before the determining, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world, the method further comprises:

in response to a system on chip comprising a plurality of processor groups, scheduling and starting, at the highest exception level, images of security services corresponding to all processor groups, and obtaining registration information of the security services, wherein the registration information comprises at least one of identifiers of the security services and processor group identifiers bound to the security services;

generating registration requests based on the registration information of the security services, and initiating, by the security services, respective registration requests to the security service scheduling module, wherein the registration requests carry the registration information of the security services; and

in response to the registration requests, generating and saving the context table by the security service scheduling module based on the registration information of the security services, wherein the context table is used for indicating mapping relationships between different groups of processors and each of the plurality of security services.

4. The method according to any one of claims 1 to 3, further comprising:

generating a second secure monitor call instruction in response to completion of execution of the target task;

in response to the second secure monitor call instruction, switching from a second current exception level of the secure world to the highest exception level, and parsing the second secure monitor call instruction at the highest exception level to obtain a task execution result of the target task; and

switching from the highest exception level back to the first current exception level of the normal world, and returning the task execution result to the normal world.

5. The method according to claim 4, wherein the parsing the second secure monitor call instruction at the highest exception level to obtain a task execution result of the target task comprises:

parsing the second secure monitor call instruction by calling a security service scheduling module at the highest exception level, to obtain the task execution result of the target task.

6. A scheduling system for a plurality of security services, comprising a plurality of processor groups configured to switch between a normal world and a secure world, wherein the normal world comprises a plurality of normal operating systems, the secure world comprises a plurality of security services, each normal operating system corresponds to a different processor group among the plurality of processor groups, and each processor group corresponds to a different security service among the plurality of security services;

a target processor among the plurality of processor groups is configured to: generate a first secure monitor call instruction based on a target task being executed by the normal world; in response to the first secure monitor call instruction, switch from a first current exception level of the normal world to a highest exception level; determine, based on the target processor that

initiates the first secure monitor call instruction, a target processor group to which the target processor belongs; determine, based on the target processor group, a target security service corresponding to the target processor group from the plurality of security services included in the secure world, wherein the target security service is used for processing the target task; and switch from the highest exception level to a target exception level of the secure world corresponding to the target security service, and start the target security service at the target exception level of the secure world to execute the target task.

7. The scheduling system according to claim 6, wherein the target processor group is configured to perform the target task in one of the plurality of normal operating systems.

8. The scheduling system according to claim 6 or 7, wherein the plurality of processor groups are further configured to run a secure monitor at the highest exception level, wherein the secure monitor is an interface between the secure world and the normal world.

9. The scheduling system according to claim 8, wherein the secure monitor comprises a security service scheduling module configured to, in response to the first secure monitor call instruction, determine, based on the target processor that initiates the first secure monitor call instruction, the target processor group to which the target processor belongs.

10. The scheduling system according to any one of claims 6 to 9, wherein the first current exception level and the target exception level are two different states of the same privilege level.

11. A scheduling apparatus for a plurality of security services, comprising:

a first generation module, configured to generate a first secure monitor call instruction based on a target task being executed by a normal world;

a first processing module, configured to, in response to the first secure monitor call instruction, switch from a first current exception level of the normal world to a highest exception level;

a first determining module, configured to determine, based on a target processor that initiates the first secure monitor call instruction, a target processor group to which the target processor belongs;

a second determining module, configured to determine, based on the target processor group, a target security service corresponding to the target processor group from a plurality of security services included in a secure world, wherein the target security service is used for processing the target task; and

a second processing module, configured to switch from the highest exception level to a target exception level of the secure world corresponding to the target security service, and start the target security service at the target exception level of the secure world to execute the target task.

12. The scheduling apparatus according to claim 11, wherein the second determining module is specifically configured to:

obtain a processor group identifier of the target processor group; and

determine, by calling a security service scheduling module at the highest exception level, a target service identifier corresponding to the processor group identifier based on a preset context table, wherein the target service identifier is used for indicating the target security service, and the context table is used for indicating mapping relationships between different groups of processors and each of the plurality of security services.

13. The scheduling apparatus according to claim 11 or 12, further comprising:

a first obtaining module, configured to: in response to a system on chip comprising a plurality of processor groups, schedule and start, at the highest exception level, images of security services corresponding to all processor groups, and obtain registration information of the security services, wherein the registration information comprises at least one of identifiers of the security services and processor group identifiers bound to the security services;

a third processing module, configured to: generate registration requests based on the registration information of the security services, and initiate, by the security services, respective registration requests to the security service scheduling module, wherein the registration requests carry the registration information of the security services; and

a fourth processing module, configured to: in response to the registration requests, generate and save the context table by the security service scheduling module based on the registration information of the security services, wherein the context table is used for indicating mapping relationships between different groups of processors and each of the plurality of security services.

14. A computer readable storage medium, wherein the storage medium stores a computer program, and the computer program is used for implementing the scheduling method for a plurality of security services according to any one of claims 1 to 5.

15. An electronic device, comprising:

a processor; and

a memory, configured to store instructions executable by the processor, wherein

the processor is configured to read the executable instructions from the memory, and execute the instructions to implement the scheduling method for a plurality of security services according to any one of claims 1 to 5; or

the electronic device comprises the scheduling system for a plurality of security services according to any one of claims 6 to 10.