> **快速链接:**
>
> 👉👉👉 ARMv8/ARMv9架构入门到精通-[目录] 👈👈👈
>
> - 付费专栏-付费课程 **【购买须知】:**
> - 联系方式-加入交流群 ----**联系方式-加入交流群**
> - 个人博客笔记导读目录(全部)

引流关键词:armv8, armv9, gic,gicv2,gicv3,异常, 中断，irq,fiq,serror,sync，同步异常，异步异常，向量表，向量表基地址，VBAR，vbar_el3，中断嵌套，中断级联，Linux Kernel，optee,ATF,TF-A,optee,hypervisor, SPM

目录

环境配置:
在linux/optee双系统环境下, linux系统的SCR.IRQ=0、SCR.FIQ=1, optee系统的SCR.IRQ=0、SCR.FIQ=0

说明:
group1是非安全中断、secure group1是安全中断

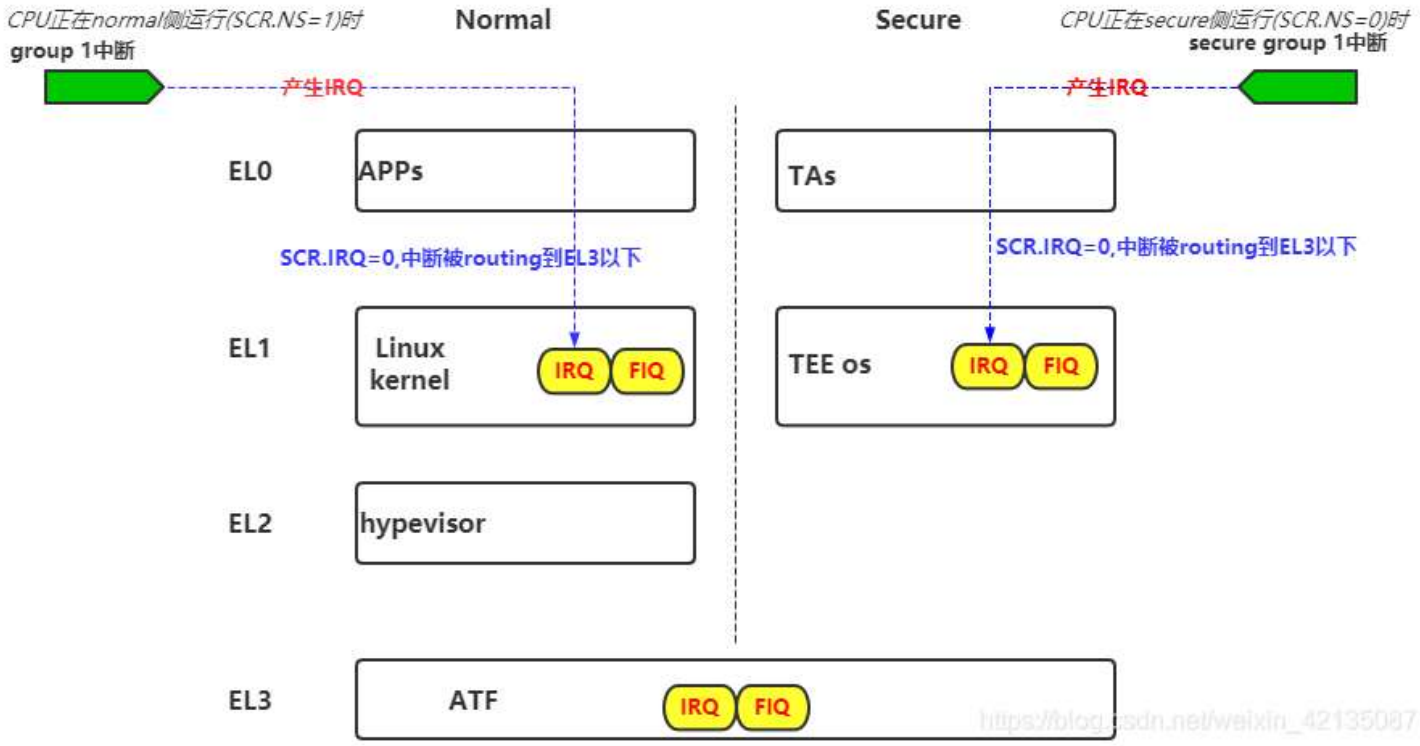
1、当cpu处于REE，来了一个非安全中断

当cpu处于normal侧时，来了一个非安全中断，根据SCR.NS=1/中断在group1组，cpu interface将会给cpu一个IRQ，(由于SCR.IRQ=0，IRQ将被routing到EL1)，cpu跳转至linux的irq中断异常向量表， 处

理完毕后再返回到normal(linux)侧.

2、当cpu处于TEE，来了一个安全中断

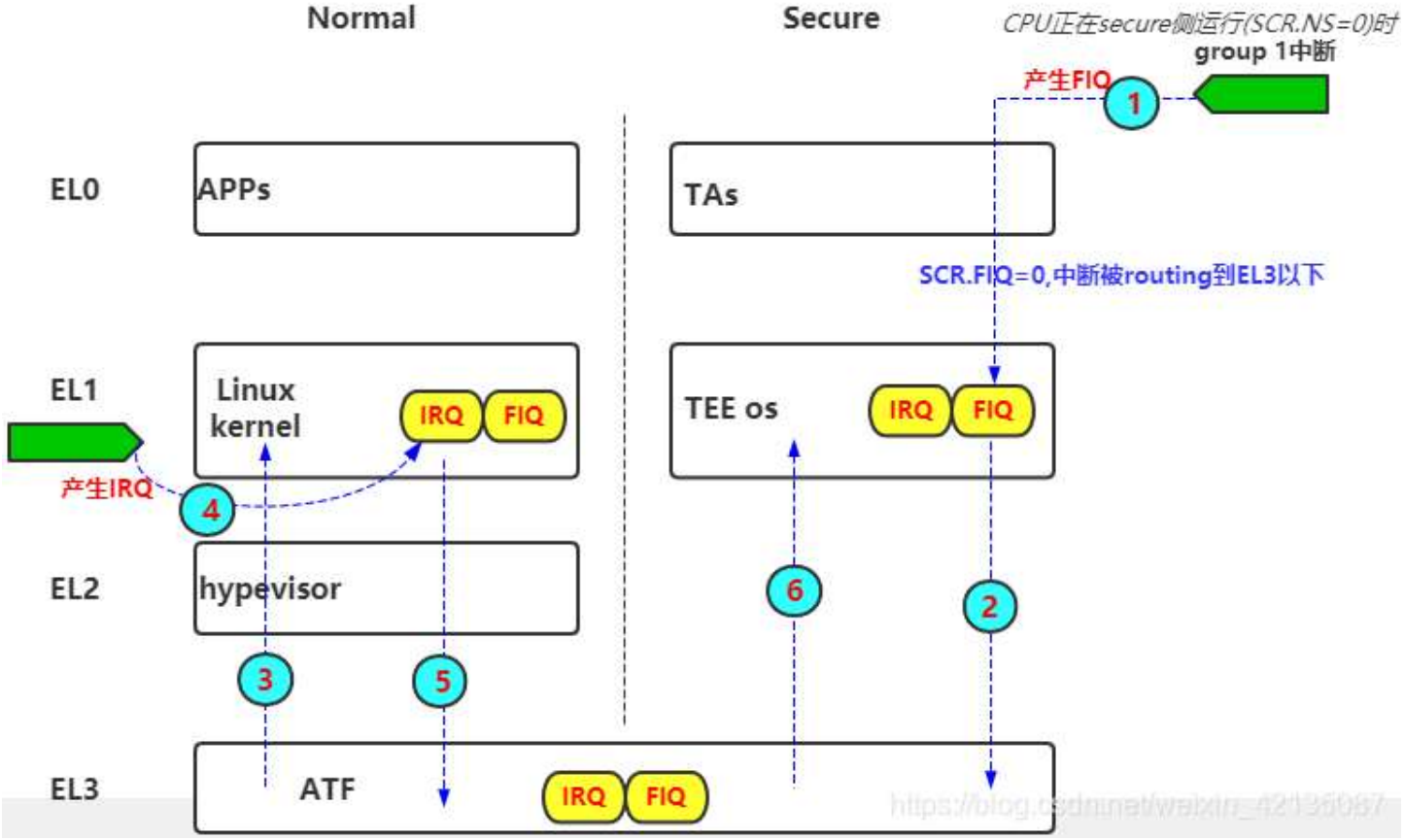
当cpu处于secure侧时，来了一个安全中断，根据SCR.NS=0/中断在secure group1组，cpu interface将会给cpu一个IRQ，(由于SCR.IRQ=0，IRQ将被routing到EL1)，cpu跳转至optee的irq中断异常向量表， 处理完毕后再返回到secure(optee)侧.



3、当cpu处于TEE，来了一个非安全中断

当cpu处于secure侧时，来了一个非安全中断，根据SCR.NS=0/中断在group1组，cpu interface将会给cpu一个FIQ，(由于SCR.FIQ=0，FIQ将被routing到EL1),跳转至optee的fiq中断异常向量表,
再optee的fiq处理函数中，直接调用了smc跳转到ATF， ATF再切换至normal EL1(linux)， 此时SCR.NS的状态发生变化，根据SCR.NS=1/中断在group1组，cpu interface会再给cpu发送一个IRQ异常，

cpu跳转至linux的irq中断异常向量表，处理完毕后，再依次返回到ATF—返回到optee

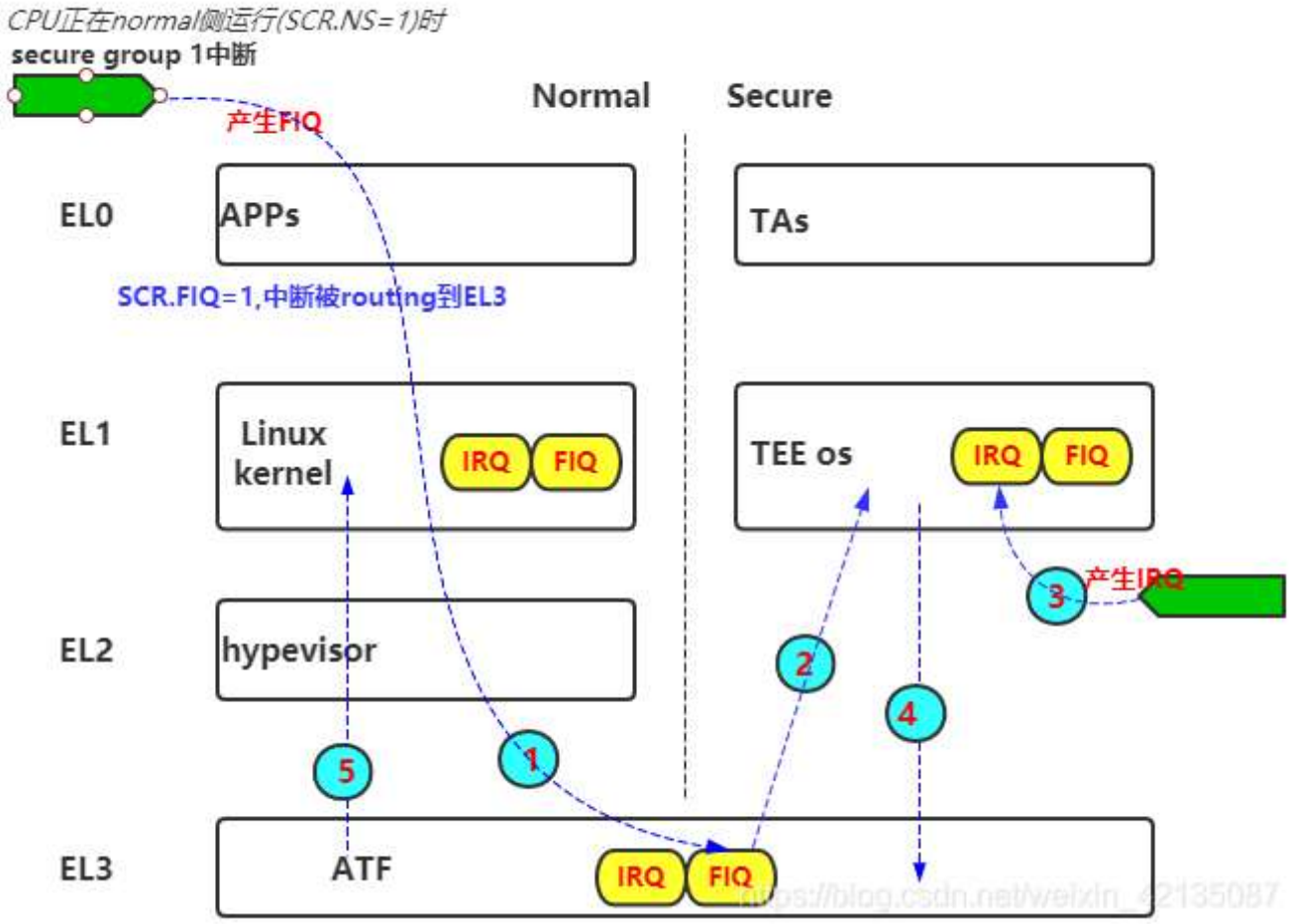


4、当cpu处于REE，来了一个安全中断

当cpu处于normal侧时，来了一个安全中断，根据SCR.NS=0/中断在group1组，cpu interface将会给cpu一个FIQ，(由于SCR.FIQ=1，FIQ将被routing到EL3),在EL3(ATF)中，判断该中断是需要optee来处理的，会切换到optee。

此时SCR.NS的状态发生变化，根据SCR.NS=0/中断在secure group1组，cpu interface会再给cpu发送

一个IRQ异常,cpu跳转至optee的irq中断异常向量表， 处理完毕后再依次返回到ATF—返回到linux



5、当cpu处于ATF时，来了一个安全中断或非安全中断(G1NS、G1S)

当cpu处于EL3时，来得任何target到EL3的中断，都将被标记位FIQ

EL and Security state of PE	Group 0	Group 1	
		Secure	Non-secure
Secure EL0/1	FIQ	IRQ	FIQ
Non-secure EL0/1/2	FIQ	FIQ	IRQ
EL3	FIQ	FIQ	FIQ

当cpu处于EL3时，配置SCR.XXX（XXX=EA或IRQ或FIQ）为0的中断不会被taken，配置SCR.XXX为1的中断将会直接target到EL3。

SCR_EL3				HCR				Target when taken from			
NS	EEL2	EA IRQ FIQ	RW	TGE	AMO IMO FMO	E2H	RW	EL0	EL1	EL2	EL3
0	0	0	0	x	x	x	x	FIQ, IRQ, Abt	FIQ, IRQ, Abt	n/a	C
0	0	0	1	x	x	x	x	EL1	EL1	n/a	C
0	0	1	x	x	x	x	x	EL3	EL3	n/a	EL3

所以在 linux系统的SCR.IRQ=0、SCR.FIQ=1, optee系统的SCR.IRQ=0、SCR.FIQ=0的场景下，总结如下，当cpu运行在EL3时：

- SCR_EL3为optee的cpu context时，来了一个G1S，中断将不会被taken
- SCR_EL3为optee的cpu context时，来了一个G1NS，中断将不会被taken
- SCR_EL3为linux的cpu context时，来了一个G1S，中断将会直接target到EL3
- SCR_EL3为linux的cpu context时，来了一个G1NS，中断将不会被taken

6、当cpu处于EL3/EL2/EL1/EL0时，来了一个ATF(group0)中断(G0)

当cpu处于EL3/EL2/EL1/EL0时，来了一个G0中断，中断将被标记位FIQ

EL and Security state of PE	Group 0	Group 1	
		Secure	Non-secure
Secure EL0/1	FIQ	IRQ	FIQ
Non-secure EL0/1/2	FIQ	FIQ	IRQ
EL3	FIQ	FIQ	FIQ

在 linux系统的SCR.IRQ=0、SCR.FIQ=1, optee系统的SCR.IRQ=0、SCR.FIQ=0的场景下，总结如下：

- 当cpu正在Non-secure EL0/1/2运行时，来了G0中断，中断被标记为FIQ，直接target到EL3
- 当cpu正在secure EL0/1/2运行时，来了G0中断，中断被标记为FIQ，中断target到了EL0/1/2，在该程序的fiq_offset会调用smc将cpu切回到EL3，到了EL3之后，中断不会被taken, 会继续返回到Non-secure EL0/1/2，然后cpu interface重新给core发送FIQ，接着又是直接target到EL3，EL3处理该中断。
- 当cpu正在EL3时，来了一个G0中断，中断会被标记为FIQ，中断target到EL3。

7、思考-中断流程举例:在TEE侧时产生了FIQ，回到REE后为啥又产生了IRQ

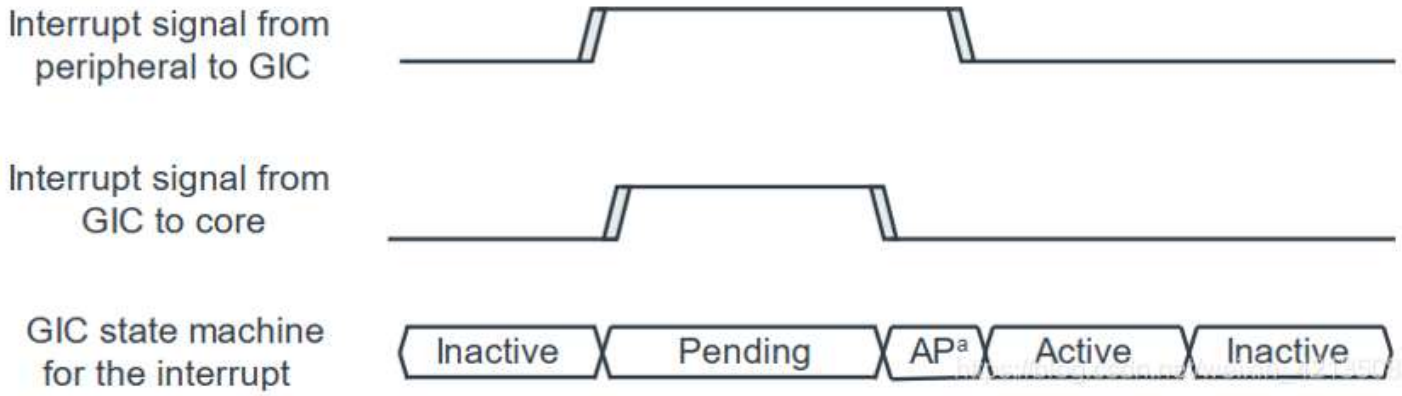
在深入研读GICV3文档后，终于找到了答案。

首先我们了解下中断优先级，在CPU interfaces (ICC_*_ELn)寄存器的描述中：

> • Provide general control and configuration to enable interrupt handling
> • Acknowledge an interrupt
> • Perform a priority drop and deactivation of interrupts
> • Set an interrupt priority mask for the PE
> • Define the preemption policy for the PE
> • Determine the highest priority pending interrupt for the PE

也就是cpu interface掌管着中断优先级和将IRQ/FIQ发送给ARM Core.

我们以Level sensitive interrupts的中断为例，先不考虑active and pending的情况：CPU interface发送给Core后，中断状态变为pending，当Core acknowledge中断后(PE跳转到中断向量表), 中断状态变为active，当中断退出后，Cpu interface会再次将优先级最高的中断发送给Core，Core处理下一个中断。



我们再看下中断的退出流程(End of interrupt), 中断的退出有两种方式：

• Priority drop 将中断优先级降到中断产生之前的值
• Deactivation 将中断从active变成inactive – (多数情况下，使用这个场景)

重点来了，在中断退出的时候，软件中一般会有Priority drop和Deactivation，既要么将中断优先级降低，要么将中断变为inactive，那么中断退出之后，cpu interface感知到的优先级最高的中断，就可能不会是此中断了，一切运行正常，符合业务…

那么我们再看下上述的中断流程举例，在TEE中，cpu interface发了一个FIQ给Core，跳转到optee的FIQ向量表，在FIQ的处理流程中，软件几乎什么都没干，没有Priority drop和Deactivation， 那么当SMC切换到了EL3之后,又退回REE后，Cpu interface感知到上一个中断处理完成，会再次发送下一个优先级最高的中断，由于之前的中断号的优先级没变，此时基本上依然是最高的优先级。此时CPU interface会再次发送该中断给Core，由于SCR.NS发生了变化，此时Cpu interface发送给Core的就变成了IRQ…

7、思考-G1NS G1S G0都有可能产生target到EL3的FIQ，如何区分?

其实在我们的linux系统的SCR.IRQ=0、SCR.FIQ=1, optee系统的SCR.IRQ=0、SCR.FIQ=0的场景下，不考虑aarch32的情况，有两种情况会产生target到EL3的FIQ：

- （1）cpu在EL0/1/2运行时，来了一个G0中断，最终CPU将会进入到EL3的向量表中的第三组向量表。

Exception taken from	Offset for exception type			
	Synchronous (including synchronous External aborts if SCR_EL3.EASE is 0)	IRQ or vIRQ	FIQ or vFIQ	SError, vSError, or Synchronous External aborts if SCR_EL3.EASE is 1 and the target is EL3
Current Exception level with SP_EL0.	0x000	0x080	0x100	0x180
Current Exception level with SP_ELx, x0.	0x200	0x280	0x300	0x380
Lower Exception level, where the implemented level immediately lower than the target level is using AArch64.	0x400	0x480	0x500	0x580
Lower Exception level, where the implemented level immediately lower than the target level is using AArch32.	0x600	0x680	0x700	0x780

- （2）ccpu在EL3运行时，来了一个G0中断，最终CPU将会进入到EL3的向量表中的第二组向量表
不过很遗憾，ATF中的向量表中未实现第二组向量表。那么为什么不需要实现呢？
在ATF/docs/firmware-design.md中找到了答案, 原来是在进入ATF之前，disable了所有的exception，ATF又没有修改PSTATE.DAIF，所有在ATF Runtime时 irq/fiq/serror/svnc都是disabled。所以异常向量表的第二行，也就用不着了。

> **Required CPU state when calling `bl31_entrypoint()` during cold boot**
>
> This function must only be called by the primary CPU.
>
> On entry to this function the calling primary CPU must be executing in AArch64 EL3, little-endian data access, and all interrupt sources masked:
>
> - PSTATE.EL = 3
> - PSTATE.RW = 1
> - PSTATE.DAIF = 0xf
> - SCTLR_EL3.EE = 0

- （3）cpu在normal EL0/1/2/3运行时(Linux侧的SCR_EL3.FIQ=1的情况下)，来了一个G1S中断，CPU将会target到EL3的向量表中的第三组向量表。

那么在ATF中第三组向量表中的fiq_offset中，是如何区分上述(1)(3)中的场景呢，即如何区分该中断是给EL3 handler处理的，还是给optee的handler处理的？ 此时1020-1023号中断发生了作用。

ID	Meaning	Example scenario
1020	Only returned by reads of `ICC_IAR0_EL1`. Highest pending interrupt is Secure Group 1. Only seen when taking FIQ to EL3	An interrupt for the Trusted OS was signaled while the PE was executing in Non-secure state. This is taken as an FIQ to EL3, so that the Secure Monitor could context switch to the Trusted OS.
1021	Only returned by reads of `ICC_IAR0_EL1`. Highest pending interrupt is Non-secure Group 1. Only seen when taking FIQ to EL3	An interrupt for the rich OS was signaled while the PE was executing in Secure state. This would be taken as a FIQ to EL3, so that the Secure Monitor could context switch to the rich OS.
1022	Used only for legacy operation.	Legacy operation is not addressed in this document.
1023	Spurious interrupt. There are no enabled INTIDs in the pending state, or all INTIDs in that pending are of insufficient priority to be taken.	When polling the IARs, this value indicates that there are no interrupts to available to acknowledge.

我们应该会用到1020，那么用在哪里的呢？请看上述汇编代码bl plat_ic_get_pending_interrupt_type的具体实现:

```
uint32_t plat_ic_get_pending_interrupt_type(void)
{
    unsigned int irqnr;

    assert(IS_IN_EL3());
    irqnr = gicv3_get_pending_interrupt_type();

    switch (irqnr) {
    case PENDING_G1S_INTID:
        return INTR_TYPE_S_EL1;
    case PENDING_G1NS_INTID:
        return INTR_TYPE_NS;

```

```
    case GIC_SPURIOUS_INTERRUPT:
        return INTR_TYPE_INVAL;
    default:
        return INTR_TYPE_EL3;
    }
}
```

其实就是在读取pending的中断号，看看有没有1020或1021，从而获得此次的中断是从secure或non-secure过来的，还是在EL3产生的。然后走相应的逻辑。

Armv8/Armv9架构从入门到精通，Armv8/Armv9架构从入门到精通（一期），Armv8/Armv9架构从入门到精通（二期）Armv8/Armv9架构从入门到精通（三期），Arm一期、Arm二期、学习资料、免费、下载，全套资料，Secureboot从入门到精通，secureboot训练营，ATF架构从入门到精通、optee系统精讲、secureboot精讲，Trustzone/TEE/安全快速入门班，Trustzone/TEE/安全标准版，Trustzone/TEE/安全高配版。全套资料。周贺贺，baron，代码改变世界，coding_the_world，Arm精选，arm_2023，安全启动，加密启动

optee、ATF、TF-A、Trustzone、optee3.14、MMU、VMSA、cache、TLB、arm、armv8、armv9、TEE、安全、内存管理、页表，Non-cacheable,Cacheable, non-shareable,inner-shareable,outer-shareable, optee、ATF、TF-A、Trustzone、optee3.14、MMU、VMSA、cache、TLB、arm、armv8、armv9、TEE、安全、内存管理、页表…