固件Runtime Server接口导致内核panic问题分析

一、分析dmesg log，找到关键错误信如下：

[    2.676149] Unable to handle kernel paging request at virtual address 0000080028001014  
[    2.684037] Mem abort info:  
[    2.686820]   ESR = 0x96000004  
[    2.689862]   Exception class = DABT (current EL), IL = 32 bits  
[    2.695755]   SET = 0, FnV = 0  
[    2.698796]   EA = 0, S1PTW = 0  
[    2.701922] Data abort info:  
[    2.704791]   ISV = 0, ISS = 0x00000004  
[    2.708601]   CM = 0, WnR = 0  
[    2.711551] [0000080028001014] user address but active\_mm is swapper  
[    2.717877] Internal error: Oops: 96000004 [#1] SMP  
[    2.722730] Modules linked in:  
[    2.725769] Process kworker/u128:8 (pid: 483, stack limit = 0x(\_\_\_\_ptrval\_\_\_\_))  
[    2.733043] CPU: 30 PID: 483 Comm: kworker/u128:8 Not tainted 4.19.0-arm64-server #1106  
[    2.741006] Hardware name: THTF CQ F520MTTY Series/THTF-FT2000Plus-64-1W-PCBVA-T1SMFT-E4, BIOS 4.0 03/27/20 14:41:09  
[    2.751481] Workqueue: efi\_rts\_wq efi\_call\_rts  
[    2.755904] pstate: 20000005 (nzCv daif -PAN -UAO)  
[    2.760671] pc : 0x204d598c  
[    2.763448] lr : 0x204d4b90  
[    2.766226] sp : ffff8080e6a2f9d0  
[    2.769522] x29: ffff8080e6a2f9f0 x28: 0000000000000000   
[    2.774811] x27: ffff8080f1c71db8 x26: ffff000008e0b840   
[    2.780096] x25: 0000000000000000 x24: 0000000000000000   
[    2.785383] x23: 0000000000000000 x22: ffff000008ed6f60   
[    2.790669] x21: ffff8080f2e8fc04 x20: ffff8080f2e8fc10   
[    2.795956] x19: ffff8080f2e8fb98 x18: 0000000000000037   
[    2.801242] x17: 0000000000000028 x16: 0000000000000000   
[    2.806529] x15: 0000000000000400 x14: 0000000000000400   
[    2.811814] x13: 0000000000000400 x12: 0000000000000030   
[    2.817101] x11: 0101010101010101 x10: 7f7f7f7f7f7f7f7f   
[    2.822388] x9 : ffff8080f0e03f60 x8 : ffff8080e6a2fa88   
[    2.827674] x7 : 0000000000000001 x6 : 0000000000000000   
[    2.832961] x5 : 0000000000000000 x4 : 0000000000000001   
[    2.838247] x3 : 0000000000000000 x2 : 0000000000000001   
[    2.843534] x1 : 0000000000000014 x0 : 0000080028001014   
[    2.848819] Call trace:  
[    2.851253]  0x204d598c  
[    2.853685]  0x204d56b4  
[    2.856118]  0x204d108c  
[    2.858551]  0x204d2280  
[    2.860983]  0x204d0c58  
[    2.863416]  0x204d0cb0  
[    2.865848]  0x204d0cf8  
[    2.868281]  0x204d0738  
[    2.870715]  0x20391874  
[    2.873148]  0x203909ac  
[    2.875584]  \_\_efi\_rt\_asm\_wrapper+0x28/0x44  
[    2.879746]  efi\_call\_rts+0xcc/0x408  
[    2.883305]  process\_one\_work+0x1f8/0x358  
[    2.887293]  worker\_thread+0x268/0x4a8  
[    2.891023]  kthread+0x118/0x128  
[    2.894235]  ret\_from\_fork+0x10/0x18  
[    2.897793] Code: d65f03c0 d10083ff f90007e0 f94007e0 (39400000)   
[    2.903857] ---[ end trace 48e34d10a00804f4 ]---

从log中可以看出来，是一个非法地址访问的panic错误，查看错误堆栈，可以分析得出这个问题是OS调用固件Runtime Server时出现的问题，所以问题是固件Bios的问题，非OS问题。

二、固件问题分析

由于堆栈是下面这种打印，我们没有办法调试固件的代码，无法反汇编调试到具体出错代码

[    2.848819] Call trace:  
[    2.851253]  0x204d598c  
[    2.853685]  0x204d56b4  
[    2.856118]  0x204d108c  
[    2.858551]  0x204d2280  
[    2.860983]  0x204d0c58  
[    2.863416]  0x204d0cb0  
[    2.865848]  0x204d0cf8  
[    2.868281]  0x204d0738  
[    2.870715]  0x20391874  
[    2.873148]  0x203909ac

通过查看无效地址，可以看到0000080028001014应该是个实地址空间，我们需要确认这个地址的含义

Unable to handle kernel paging request at virtual address 0000080028001014

通过查看飞腾服务器手册，可以看到此地址空间是串口寄存器的实地址空间



所以问题根因找到了固件的Runtime Server接口有打印信息，并且BIOS中并未对此地址并没有进行虚拟地址转化。OS在调用Runtime接口的时，触发了BIOS里面的打印，造成了实地址访问，导致发生非法地址访问的Panic问题。

客户更新固件后此问题得到解决。

3、问题总结

a、遇到efi\_call\_rts错误堆栈，此类问题基本都是固件Bios问题，非OS问题

b、rts是Bios提供给OS通过uefi调用的接口，bios会在jump到os之前把关键的变量进行虚拟地址转化。相关流程和原理内核开发人员需要熟悉。

c、芯片寄存器地址可以通过查看文档的方式获取，也可以通过解析固件的ACPI表来获取。详细操作过程见附录

四、附录 固件ACPI表信息解析

1、安装iasl解析工具

uos@uos-PC:~$ sudo apt install acpica-tools

正在读取软件包列表... 完成

正在分析软件包的依赖关系树

正在读取状态信息... 完成

acpica-tools 已经是最新版 (20181213-1)。

下列软件包是自动安装的并且现在不需要了：

imageworsener libmaxminddb0 libqtermwidget5-0 libutf8proc2 qtermwidget5-data

使用'sudo apt autoremove'来卸载它(它们)。

升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 5 个软件包未被升级。

2、拷贝出设备的DSDT文件，并修改权限

uos@uos-PC:~$ sudo cp /sys/firmware/acpi/tables/DSDT ~/DSDT

uos@uos-PC:~$ sudo chmod 777 DSDT

3、拷贝出设备的DSDT文件，并修改权限

uos@uos-PC:~$ iasl -d DSDT

Intel ACPI Component Architecture

ASL+ Optimizing Compiler/Disassembler version 20181213

Copyright (c) 2000 - 2018 Intel Corporation

Input file DSDT, Length 0x1059 (4185) bytes

ACPI: DSDT 0x0000000000000000 001059 (v02 ARMLTD PHYTIUM 20180509 INTL 20180105)

Pass 1 parse of [DSDT]

Pass 2 parse of [DSDT]

Parsing Deferred Opcodes (Methods/Buffers/Packages/Regions)

Parsing completed

Disassembly completed

ASL Output: DSDT.dsl - 34287 bytes

4、搜索需要关注的信息，比如串口信息，可以看到串口寄存器的基地址、中断号等信息

Device (COM0)

{

Name (\_HID, "HISI0031") // \_HID: Hardware ID

Name (\_UID, Zero) // \_UID: Unique ID

Method (\_CRS, 0, Serialized) // \_CRS: Current Resource Settings

{

Name (SER0, ResourceTemplate ()

{

QWordMemory (ResourceProducer, PosDecode, MinFixed, MaxFixed, NonCacheable, ReadWrite,

0x0000000000000000, // Granularity

0x0000080028001000, // Range Minimum

0x0000080028001FFF, // Range Maximum

0x0000000000000000, // Translation Offset

0x0000000000001000, // Length

,, \_Y00, AddressRangeMemory, TypeStatic)

Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive, ,, )

{

0x00000043,

}

})

CreateQWordField (SER0, \\_SB.COM0.\_CRS.\_Y00.\_MIN, U0BL) // \_MIN: Minimum Base Address

CreateQWordField (SER0, \\_SB.COM0.\_CRS.\_Y00.\_MAX, U0BH) // \_MAX: Maximum Base Address

U0BL = 0x0000080028001000

U0BH = 0x0000080028001000

U0BH += 0x0FFF

Return (SER0) /\* \\_SB\_.COM0.\_CRS.SER0 \*/

}