DragonFlyBSD - Bug #2013

oversized DMA request loop

03/04/2011 12:19 PM - josepht

Status:	In Progress	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	50%
Category:	Kernel	**Estimated time:**	0.00 hour
Target version:	4.2.x		

Description

I am getting an oversized DMA transfer request loop when attempting to
play a DVD with mplayer.

Mar 3 20:19:50 neptune kernel: acd0: setting up DMA failed
Mar 3 20:19:50 neptune kernel: ata1: FAILURE - oversized DMA transfer
attempt 73728 > 65536
Mar 3 20:19:50 neptune kernel: acd0: setting up DMA failed
Mar 3 20:19:50 neptune kernel: ata1: FAILURE - oversized DMA transfer
attempt 73728 > 65536

I instrumented the kernel with some kprintf()s and a panic() in an
attempt to track this down. I've upload a coredump to
leaf:~josepht/crash/dvd_crash.tgz. I've gone back as far as
fe667cd2edce8ea4f3dcc86526ed9759c14f0bf4 but haven't yet been able to
track down exactly when this started happening. I was not seeing this
on a system built around Jan. 27 2011, though I'm not certain how old
the sources were prior to that build.

Here's the backtrace. Note this panic() was manually inserted.

Unread portion of the kernel message buffer:
ata1: FAILURE - oversized DMA transfer attempt 73728 > 65536
panic: JAT: panic
cpuid = 0
Trace beginning at frame 0xda2f18c0
panic(ffffffff) at panic+0x164
panic(c05cffe9,c4747878,c0609488,12000,10000) at panic+0x164
ata_dmaload(c4747878,d2afe800,12000,2,d7110000) at ata_dmaload+0x9e
ata_begin_transaction(d9c4b788) at ata_begin_transaction+0x33b
ata_start(c4747878) at ata_start+0x1ef
ata_queue_request(d9c4b788) at ata_queue_request+0x44a
atapi_action(c4786ce8,d6a084e0,c46a4310,c49825e8,c4623f40) at
atapi_action+0x815
xpt_run_dev_sendq(c0176f2b,20,1,0,a) at xpt_run_dev_sendq+0x1a7
xpt_action(d6a084e0) at xpt_action+0x25b
cdstart(d69f25e0,d6a084e0,d6a084e0,c4623f40,1,c498261c,1) at
cdstart+0x219
xpt_run_dev_allocq(d6c38dd0,d69f25e0,c49f79d0,da2f1ae8,c0176e39) at
xpt_run_dev_allocq+0x70
xpt_schedule(d69f25e0,1) at xpt_schedule+0xec
cdstrategy(da2f1af4) at cdstrategy+0xd2
dev_dstrategy(d6b5cb10,c49f7a50) at dev_dstrategy+0xed
dsched_strategy_raw(d6c38e0c,c49f7a50,da2f1b5c,c0325612,c49b1800) at
dsched_strategy_raw+0x41
noop_queue(c49b1800,d9dda700,c49f7a50,c49b1800) at noop_queue+0x11
dsched_queue(d6c38e0c,c49f7a50) at dsched_queue+0xfe
diskstrategy(da2f1b80) at diskstrategy+0x55
dev_dstrategy(d6b5cbf0,c49f7a00,d6b5cbf0,0,0) at dev_dstrategy+0xed
physio(da2f1c00,c02fbc61,da2f1be8,c0693f38,d6b5cbf0) at physio+0x1d8
physread(da2f1be8,c0693f38,d6b5cbf0,da2f1ca0,0) at physread+0x14
dev_dread(d6b5cbf0,da2f1ca0,0,d9d0acd8,0) at dev_dread+0x6c
devfs_fo_read(d4ef2320,da2f1ca0,d4e941d8,0,d4ef2320) at

```
devfs_fo_read+0x150
kern_preadv(3,da2f1ca0,0,da2f1cf0,292a2800) at kern_preadv+0xfb
sys_read(da2f1cf0,1e99f,0,d6a0dee0,200282) at sys_read+0x5d
syscall2(da2f1d40) at syscall2+0x240
Xint0x80_syscall() at Xint0x80_syscall+0x36
Debugger("panic")
```

Here's the diff from master as of Thu Mar 3 22:33:09 EST 2011

```
diff --git a/sys/dev/disk/nata/ata-chipset.c b/sys/dev/disk/nata/ata-chipset.c
index 91d4444..b155bf1 100644
--- a/sys/dev/disk/nata/ata-chipset.c
+++ b/sys/dev/disk/nata/ata-chipset.c
@@ -892,6 +892,7 @@ ata_ahci_dmainit(device_t dev)
/* note start and stop are not used here */
ch->dma->setprd = ata_ahci_dmasetprd;
ch->dma->max_iosize = 8192 * DEV_BSIZE;
+ kprintf("JAT: %s\n", __func__);
if (ATA_INL(ctlr->r_res2, ATA_AHCI_CAP) & ATA_AHCI_CAP_64BIT)
ch->dma->max_address = BUS_SPACE_MAXADDR;
}
@@ -1531,6 +1532,7 @@ ata_cyrix_setmode(device_t dev, int mode)

ch->dma->alignment = 16;
ch->dma->max_iosize = 126 * DEV_BSIZE;
+ kprintf("JAT: %s\n", __func__);

mode = ata_limit_mode(dev, mode, ATA_UDMA2);

@@ -3071,6 +3073,7 @@ ata_marvell_edma_dmainit(device_t dev)

/* chip does not reliably do 64K DMA transfers */
ch->dma->max_iosize = 126 * DEV_BSIZE;
+ kprintf("JAT: %s\n", __func__);
}
}

@@ -3121,6 +3124,7 @@ ata_national_setmode(device_t dev, int mode)

ch->dma->alignment = 16;
ch->dma->max_iosize = 126 * DEV_BSIZE;
+ kprintf("JAT: %s\n", __func__);

mode = ata_limit_mode(dev, mode, ATA_UDMA2);

@@ -4454,6 +4458,7 @@ ata_serverworks_allocate(device_t dev)
/* chip does not reliably do 64K DMA transfers */
if (ch->dma)
ch->dma->max_iosize = 126 * DEV_BSIZE;
+ kprintf("JAT: %s\n", __func__);

return 0;
}
diff --git a/sys/dev/disk/nata/ata-dma.c b/sys/dev/disk/nata/ata-dma.c
index 796eec1..bc0d841 100644
--- a/sys/dev/disk/nata/ata-dma.c
+++ b/sys/dev/disk/nata/ata-dma.c
@@ -76,6 +76,7 @@ ata_dmainit(device_t dev)
ch->dma->boundary = 128 * DEV_BSIZE;
ch->dma->segsize = 128 * DEV_BSIZE;
ch->dma->max_iosize = 128 * DEV_BSIZE;
+ kprintf("JAT: %s\n", __func__);
ch->dma->max_address = BUS_SPACE_MAXADDR_32BIT;
}

@@ -238,6 +239,7 @@ ata_dmaload(device_t dev, caddr_t data, int32_t count, int dir,
if (count > ch->dma->max_iosize) {
```

```
device_printf(dev, "FAILURE - oversized DMA transfer attempt %d > %d\n",
count, ch->dma->max_iosize);
+ panic("JAT: panic");
return EIO;
}

diff --git a/sys/dev/disk/nata/atapi-cd.c b/sys/dev/disk/nata/atapi-cd.c
index 12926ba..f2fc2f9 100644
--- a/sys/dev/disk/nata/atapi-cd.c
+++ b/sys/dev/disk/nata/atapi-cd.c
@@ -956,8 +956,10 @@ acd_set_ioparm(device_t dev)
struct acd_softc *cdp = device_get_ivars(dev);
struct disk_info info;

-    if (ch->dma)
+    if (ch->dma) {
cdp->iomax = min(ch->dma->max_iosize, 65534);
+ kprintf("JAT: %s\n", __func__);
+ }
else
cdp->iomax = min(DFLTPHYS, 65534);

Thanks,
Joe
```

History

#1 - 05/25/2014 07:24 AM - zcrownover

- Description updated

- Category set to Kernel

- Target version set to 3.6.1

I'm running 3.6.2 and seeing the same error, although the size of the message is a little smaller, it's causing HAMMER to peg the CPU while it handles 609 additions of that to the kernel messages per second.

#2 - 05/26/2014 11:05 AM - zcrownover

I had this error in a Virtual Box VM that had a corrupted ISO attached to it, when I removed the ISO, the error stopped entirely. If this matches your scenario, let me know so we can close this out.

#3 - 05/31/2014 05:06 PM - zcrownover

- Target version changed from 3.6.1 to 3.8.0

Running 3.8.0rc and this error is occuring with no disk in and flooding syslog still.

```
> sudo tail -n 20 /var/log/messages
Password:
May 31 17:03:39 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:39 zach-dfly kernel: acd0: setting up DMA failed
May 31 17:03:40 zach-dfly last message repeated 2114 times
May 31 17:03:40 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:40 zach-dfly kernel: acd0: setting up DMA failed
May 31 17:03:41 zach-dfly last message repeated 2114 times
May 31 17:03:41 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:41 zach-dfly kernel: acd0: setting up DMA failed
May 31 17:03:42 zach-dfly last message repeated 2120 times
May 31 17:03:42 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:42 zach-dfly kernel: acd0: setting up DMA failed
May 31 17:03:43 zach-dfly last message repeated 2108 times
May 31 17:03:43 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:43 zach-dfly kernel: acd0: setting up DMA failed
May 31 17:03:44 zach-dfly last message repeated 1723 times
May 31 17:03:44 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:44 zach-dfly kernel: acd0: setting up DMA failed
May 31 17:03:45 zach-dfly last message repeated 1902 times
May 31 17:03:45 zach-dfly kernel: ata1: FAILURE - oversized DMA transfer attempt 69632 > 65536
May 31 17:03:45 zach-dfly kernel: acd0: setting up DMA failed
>
```

I'm about to upgrade this system to 3.8.0rc2, though I don't think any of the changes made from rc1 to rc2 would affect this. For reference, although Virtualbox claims are often dismissive as I don't know if this is due to Virtualbox or not and don't have this issue on other instances and most of my other instances are on Virtualbox as well. My Virtualbox version with this instance is 4.3.12 r93733

#4 - 06/01/2014 06:09 AM - swildner

- Assignee deleted (0)

Does it help if you set hw.ata.atapi_dma=0 in /boot/loader.conf?

#5 - 06/01/2014 09:47 AM - zcrownover

- % Done changed from 0 to 100

Setting that seems to have silenced the messages dumped to the kernel and also caused the system to be stable.

#6 - 01/14/2015 03:48 PM - tuxillo

- Status changed from New to In Progress

- Target version changed from 3.8.0 to 4.2.x

- % Done changed from 100 to 50

Hi,

Disabling DMA isn't a solution I would say.

Cheers,
Antonio Huete