H168NV20拷机VDSL上行STA掉线无法重新关联

【作者】 李瑞，荆杰，邵存金

【时间】 2012-12-06

【摘要】 本文详细分析了H168NV20拷机STA掉线无法重新关联的查障流程，记录故障查证的思路和解决方法。通过本文，可以对其它单板无线故障的查证流程和方法提供借鉴和经验。

一、故障背景

故障描述：

【H168N V20】拷机测试STA掉线后，无法重新关联；

测试环境：

1. 单板建立一条pppoe拨号wan连接，开启ssid1，开启igmp_proxy，开启igmp_snooping；

2. LAN侧4路PC播放iptv，2台以上sta关联ssid1，bt下载；

现象：

拷机2小时候，单板无任何异常打印，CPU占用率3.8%左右，nf_contract数1000条左右；eapd，nas等进程正常，STA无法关联单板；

二、查证流程及思路

2.1、常规武器——空口抓包

STA无法关联modem故障定位的通用步骤：第一步：查看单板CPU占用率是否达到99.9%，nf_contrack条目数是否达到最大；第二步：禁用，启用STA无线网卡，看是否能关联；第三步：禁用、启用STA无线网卡后，仍无法关联，观察STA关联单板时串口打印，空口抓包，分析关联的握手是否完成；

分析H168NV20拷机，STA无法关联单板故障，第一步发现CPU占用率很低，nf_contract也很低；第二步禁用启用STA无线网卡，依然无法关联；执行第三步，发现串口没有打印关联握手时的req，response，空口抓包，发现STA发出的AUTH报文，单板没有回复相应的AUTH success报文；判断STA无法关联是由于AUTH握手未正常完成导致；

2.2、查证关键——走查代码，制作debug版本，复现并定位故障

1. 通过走查代码，跟踪无线收包的流程，对比H368N,H168N_UNICOM版本无线驱动，未发现异常；

2. 跟踪无线收包流程，在DMA_RX和无线收包wlc_recv函数中增加打印，在收包路径的所有异常分支增加打印，制作debug版本；

3.搭建环境，拷机复现故障：3路IPTV+4路BT下载；

2.3、故障分析

【2012年12月08日下午五点十分左右】

这个问题当时测试部赵冲在拷机测试的时候，也出现类似故障，当时马哥以及陈振都在，我空口抓包发现sta有向单板发Auth包，但是单板没有回auth包，这个和李瑞确认了下，认定是同样现象。当时测试部赵冲的测试软件/硬件以及测试环境如下：

1、测试软件版本：H168N V2.0T3；硬件版本：Hardware Version V2.01 绿色的

Linux version 2.6.30 (xialei@localhost.localdomain) (gcc version 4.4.2 (Buildroot 2010.02-git)) #2 SMP PREEMPT Mon Nov 26 23:14:52 CST 2012

2、上行方式：VDSL

3、建链速率：58.6M/97.7M

4、WAN连接配置：

（1）Internet--VLAN 110--PPPOE路由连接（用于外网拨号)

（2）IPTV_route--VLAN 300--DHCP路由连接（用于IPTV播放）

（3）IPTV_br--VLAN 300--桥（用于IPTV播放）

5、端口绑定：

（1）Internet连接绑定SSID：SSID1、SSID2、SSID3和SSID4

（2）IPTV_route绑定LAN1和LAN2口

（3）IPTV_br绑定LAN3和LAN4口

6、开启SSID：SSID1、SSID2、SSID3和SSID4，使用默认配置进行拷机

7、拷机IPTV：4路组播，IPTV_route承载1路10M和1路12M片源，IPTV_br承载1路10M和1路12M片源

8、关联无线STA数：一共2个，均进行BT下载，下载速率和为400KB/S左右9、无其他上网业务

9、在拷机的同时，有跑脚本（主要是top free 以及查看连接数）

同时，我们自己在测试部VD环境下进行同样拷机，拷机的软件版本是李瑞添加打印调试的版本，测试软件/硬件以及环境信息如下：

1. 测试软件版本：李瑞在SVN最新库里最新代码添加打印调试后的版本；
2. 硬件版本：H168NV1.0（标号刘博 11），蓝色的板子
3. 拷机环境和赵冲的拷机环境类似（只是ssid名以及带宽不一样，我们是40Mhz下）

【2012年12月10日星期一】

鉴于12月8号测试部赵冲复现了一次该问题，今天观察周六（12月8号）李瑞测试版本的拷机情况发现，又出现该问题。抓包分析，和前面的故障现象类似。

上午和邵总又重新让赵冲帮我们搭了下周六他拷机的环境我们又做了对比测试，

第一次测试软件硬件/环境如下：

1. 测试软件：李瑞的Debug版本
2. 测试硬件：H168NV2.0，绿色的板子
3. 拷机环境由赵冲搭建，和他周六的一样，只是我们用的是在40Mhz下拷机的。

很快，大概两三个小时，问题复现。抓包显示一样。

为了验证以及复现查证该问题，又进行第二次对比测试，测试环境如下：

1测试软件：邵总对比南京H168N_UNICOM的无线驱动后，编了新版本

2.测试硬件：H168NV2.0，绿色的板子

1. 拷机环境：也是由赵冲帮忙搭建，是在无线带宽40Mhz下拷机的

中午拷机，下午很快也复现了该问题，抓包显示问题一样。

晚上，邵总为了进一步对比测试，又从库里直接取出南京的H168N_UNICOM版本，进行再次对比测试验证，验证情况如下：

1. 软件版本：\\10.46.105.2\zxv10ftp2\ZXHNDailyBuild\ZXHN H168N\H168N_UniCom_arch\Arch_images\2012-10-31\H168N-V1.1.0_ET3\chip_broadcom_h168n.2873 comp.1945 csp.2847\release中取的最新版本
2. 硬件版本：H168NV2.0 绿色版本
3. 测试环境，和赵冲前面测试环境一模一样，进行拷机测试。

结果：从12月10晚上6点30左右-12月11日上午9:30左右复现该问题，bt下载没有速度，开机的三台sta均ping不通单板，三台笔记本显示已连接状态，但是ping不通，而且断开之后均不能关联单板，串口没有异常打印，nas等进程均在。空口抓包发现问题和前面几次一样。抓包情况如下：

H168N_UNICOM20121210_20MHZ.pkt

【2012年12月12日下午五点十分左右】

复现故障后，通过走查前期添加的打印分析：

1. 添加打印的函数调用路径：wlc_bmac_recv -> wlc_recv -> wlc_recvctl ；
2. 通过查看log，发现故障复现时，无任何添加的打印，分析无线收包时未执行_dma_rx函数，该函数由无线软中断触发，因此，无线软中断存在问题，下一步分析软中断的处理流程；
3. 通过走查代码，wlc_bmac_recv函数的调用路径为：wl_isr -> wl_dpc -> wlc_dpc -> wlc_bmac_recv；通过串口查看硬中断brcm15（无线中断号）；发现硬中断数在增加；分析可能是由于软中断存在异常导致无线无法收包；
4. 在无线软中断函数调用路径中增加打印，调用路径：wl_isr -> wlc_isr -> wlc_intstatus，拷机复现，定位软中断故障点；

拷机测试环境同上；

【2012年12月13日下午八点十分左右】

H168NV20 无线拷机单板无线驱动无法收包故障查证进展：

1.添加打印函数：

跟踪无线驱动收包路径增加打印，路径：

wl_isr（wl_linux.c）-> wlc_isr（wlc_bmac.c）-> wlc_intstatus（wlc_bmac.c）-> wl_dpc（wl_linux.c）-> wlc_dpc（wlc_bmac.c）-> wlc_bmac_recv（wlc_bmac.c）-> wlc_recv（wlc.c）-> wlc_recvctl（wlc.c）

2.拷机复现故障，单板无线驱动无法收包后的串口log如下：

H168NV20_jingjie_Debug_Version20121213.log

3.重启单板后，驱动可以正常收包的log如下：

H168NV20_jingjie_Debug_Version20121213_2.log

4.对比正常和异常时的代码，参考打印跟踪收包流程，

1）正常情况下，函数wlc_intstatus获取的macintstatus = R_REG(osh, &regs->macintstatus)，经过位操作后的运算结果为0x8000；

对应的中断为：

#define MI_DMAINT (1 << 15) /* (ORed) DMA-interrupts */ /*0x8000*/

在wlc_dpc函数中，进入相应的分支进行处理：-> wlc_bmac_recv -> _dma_rx；完成收包流程；

2）异常情况下，从log可以得到，函数wlc_intstatus获取的macintstatus = R_REG(osh, &regs->macintstatus)，

经过位操作后的运算结果为0x80000， 0x40000，0x40；对应的中断为：

#define MI_PMQ (1 << 6) /* PMQ entries available */ /*0x40*/

#define MI_BG_NOISE (1 << 18) /* MAC has collected background noise samples */ /*0x40000*/

#define MI_DTIM_TBTT (1 << 19) /* MBSS DTIM TBTT indication */ /*0x80000*/

没有产生DMA-interrupts，所以无法进入收包流程，无法完成收包；

【2012年12月14日】

在BRCM官网提交CSP:599453 4路iptv+3台STA bt下载拷机10几个小时，出现STA无法关联AP。

【2012年12月17日---20日】

使用BRCM提供的patch，进行拷机验证，发现问题仍然存在。

【2012年12月25日】

拿到BRCM最新的无线驱动，准备升级驱动后验证该问题。

【2012年12月26日---31日】

升级无线驱动，解决编译问题。

【2013年1月1日---4日】

元旦期间，拷机故障未复现。

【2013年1月5日】

利用工作日拷机一天，今天早上去看，发现不收包问题复现。升级驱动加上BRCM提供的patch也未能解决该问题。

【2013年1月7日—1.11】

驱动代码已经分析了原因是由于无线驱动收包中断没有产生所致。

期间，跟超哥交流了下这个问题，超哥建议我们把wl macreg命令打开，这样可以在出故障的时候查看rx相关寄存器的值。

将驱动中的wl macreg命令打开，编出版本调试，比如：

wl macreg 0x24 4 //查看掩码寄存器0x24寄存器的值。

0x10000

wl macreg 0x24 4 0x0000//修改掩码寄存器0x24寄存器的值。

0x10000

走查代码，发现跟无线收包相关的寄存器定义如下

wl_isr()

|--wlc_isr()

|-- macintstatus = wlc_intstatus(wlc, TRUE); //read and clear macintstatus and intstatus registers

```
/* MI_DMAINT is indication of non-zero intstatus */
if (macintstatus & MI_DMAINT) {
    if(RxDebug)
        printk("wl%d:  %s %d macintstatus: 0x%x\n", wlc_hw->unit, __func__, __LINE__,macintstatus
    if (D11REV_IS(wlc_hw->corerev, 4)) {
        intstatus_rxfifo = R_REG(osh, &regs->intctrlregs[RX_FIFO].intstatus);
        intstatus_txsfifo = R_REG(osh,
                                  &regs->intctrlregs[RX_TXSTATUS_FIFO].intstatus);

        WL_TRACE(("wl%d: intstatus_rxfifo 0x%x, intstatus_txsfifo 0x%x\n",
                  wlc_hw->unit, intstatus_rxfifo, intstatus_txsfifo));
        if(RxDebug)
            printk("wl%d:  %s %d intstatus_rxfifo: 0x%x\n", wlc_hw->unit, __func__, __LINE__,ints
        /* defer unsolicited interrupt hints */
        intstatus_rxfifo &= DEF_RXINTMASK;
        intstatus_txsfifo &= DEF_RXINTMASK;
```

rx状态寄存器对应的偏移量是0x20, 掩码寄存器对应的偏移量是0x24。

在办公位测试发现，RX掩码寄存器(0x24)的值初始值是0x10000，如果修改该寄存器的值为0x0000，此时关联上AP的STA ping不通。再修改为初始值0x10000，STA可以ping通AP。也就是说，该寄存器的值可以控制AP收包。于是怀疑代码中某个地方会修改这个寄存器的值。顺着这个思路，搜索代码，发现该寄存器只有一个地方会写，就是在wlc_coreinit()函数中。

```
/* write interrupt mask */
W_REG(osh, &regs->intctrlregs[RX_FIFO].intmask, DEF_RXINTMASK);
if (D11REV_IS(wlc_hw->corerev, 4))
```

写RX掩码寄存器的值为0x10000

从无线驱动代码里看是没有地方修改rx掩码寄存器的值的。难道我们碰到的问题就是这个值被修改了？或者是无线芯片异常修改了这个值？

于是用支持wl macreg命令的版本拷机验证。故障出现时，发现rx掩码寄存器的值没有改变

wl_dpc()

|-- wlc_intrsupd()

|-- macintstatus = wlc_intstatus(wlc, FALSE);

|-- wlc->macintstatus |= macintstatus;

|-- wlc_dpc()

|--wlc_bmac_recv()

|--dma_rx()

【2013.1.14】

今天查看了拷机结果，问题复现，在串口执行wl macreg 0x24 4命令读取rx掩码寄存器的值，发现是0x10000，这个值没有被改变。

将我们这几天的分析反馈给BRCM，BRCM回复。

Has the dma been refilled?

If you do "wl dump dma", and found rx dma chain is empty. There is no dma interrupt incoming.

The root cause is that.....allocated buf is leaked someone else. Buffer pool runs out. When you try to refill rx dma with rx bd + rx buf, and found there is no allocated buffer. Your rx dma chain will be empty. So no interrupt will come from wifi mac.

--Leon

根据BRCM的提示，确认rxfill函数的调用，在驱动中相关位置添加打印信息，并且将wl dump dma命令打开，编译版本验证。

wlc_bmac_watchdog ()//bmac watchdog每隔一段时间运行

|--dma_rxfill()

|--_dma_rxfill

|--PKTGET ()//从rx ctfpool中取skb

|--osl_pktfastget()//这里会判断，当ctfpool中没有可用的skb时，会返回NULL。

故障复现时，通过wl dump dma命令可以看到此时没有可用的rx chain，也就是说dma_rxfill从rx ctfpool中取skb返回NULL，表明此时，ctfpool满了，有可能其他地方占用的这些skb，无线驱动还没有回收。跟BRCM的预测一致。

【2013.1.14】

再次将我们的分析反馈给BRCM，BRCM今天回复如下：

PKTGET

PKTGET will be invoked in refill function. If Preallocated pkt pool is enable, PKTGET have to check CTF POOL to pick up preallocated pkt from the pool(fast get). If disable, have to pick up from kernel.

The problem is that, when pool is enabled, and pool runs out. There is no more free pkt available. So rx dma chain is empty.

Basically, driver will get pkt from pool to receive, then sendup to kernel or other drivers. When incoming pkt is done, have to free pkt, which will be recycled to pool again. If the pkt is missed, it will never go back to pool. Pool will dry gradually.

--Leon

BRCM的分析跟我们的一致，继续咨询BRCM，看他们是否有比较好的方法解决该问题或者能否提供patch。BRCM回复如下：

You'd better to check whether kernel/ether driver/xtm driver is losing wifi pkt in your own modification.

1) release code hasn't this kind of issue, that means kernel/ether driver/xtm are ok in that check point.

2) the most possible reason is own modification leading to lose pkt.

3) you may think there is no issue via ethernet. Since most traffics go through fap instead of kernel.

No patch about that.

--Leon

BRCM怀疑是我们自己的修改导致该问题，看来只有我们继续分析。

上午开发经理邀请协议栈、小系统的同事一起头脑风暴，想下下一步该怎么分析该问题。讨论决定向rx预分配skb的dev name打印出来。

下午重点走查了rx ctfpool的工作原理。

wl_attach()

|--osl_ctfpool_init()//ctfpool初始化

|--osl_ctfpool_add()//预分配申请256个SKB

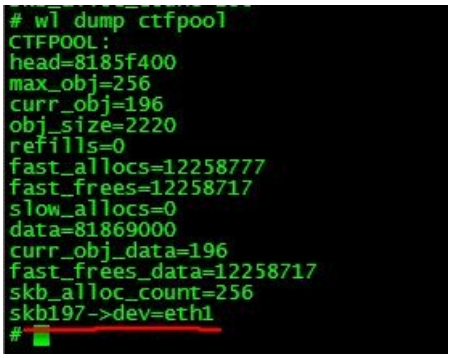
|--dev_alloc_skb()

从之前的分析，我们可以知道，dma_rxfill会从ctfpool取skb用，而ctfpool初始化的时候已经预分配好的skb，当ctfpool中的skb被占用时，dma_rxfill取不到可用的skb，就不会产生收包中断，导致故障出现。

【2013.1.15】

今天和超哥继续交流这个问题，在无线驱动中，超哥增加了wl dump ctfpol的调试命令，可以将ctfpool的相关字段打印出来，我在这个命令的基础上将

预分配skb的dev name打印出来，可以确定出故障时，哪个接口占用了ctfpool中skb。期间解决了打印skb dev name挂死的问题。



和超哥交流下这个想法，超哥怀疑：“我认为是特定类型的报文造成的，并且是wlan接收到的。建议抓下报文分析下吧，并主动构造些报文向AP发送。按你们描述的场景，IMGP报文可能需要关注下。”

将该版本在实验室进行拷机。制作脚本让wl dump ctfpol命令每隔1分钟执行一次。

【2013.1.16】

昨晚拷机未复现故障。但从wl dump ctfpool命令的打印来看，skb dev name大部分时候是ptm。

三、总结