毓杰，您好：

如刚才电话沟通，目前针对伍玉霞.dmp 的分析已完成，已找到发生蓝屏问题的原因，经您同意，此 case 将暂做关闭处理，以下为案例总结，请您知悉：

Case No：CAS-02903-W1P8K5

问题描述：

==========

CMGE 在使用无线网络过程蓝屏。

问题分析：

==========

此次蓝屏 Dump 的 Bugcheck 为 0x50（PAGE_FAULT_IN_NONPAGED_AREA），与之前 memory-copy.dmp 并不完全相同，虽然不是由于 special pool 引发的 crash，但通过 NBLTracking，我们还是得到了部分额外信息，为方便理解，以下将着重分析 vwifimf.sys 在处理 802.1x 数据包过程中的代码流程及逻辑，code 部分的修改及完善，需要 TMS 进行。

1，从 call stack 得知，Crash 蓝屏发生在网卡返回发送完成的通知后，即最终由 ndis!ndisCallSendCompleteHandler 调用各层驱动处理 exception

```
kd> dt NDIS_EXPAND_STACK_CALLOUT_PARAMS 0xffff9704`0d757190
   +0x000 Status           : 0n659
   +0x008 FilterHandle     : 0xffffc082`207db250 Void //vwifimf.sys
   +0x010 FilterContext    : 0xffffc082`1fc13d60 Void
   +0x018 CancelId         : 0xfffff800`5f4b1e28 Void
   +0x018 Request          : 0xfffff800`5f4b1e28  Unknown value 0x40f98b48
   +0x018 NetDevicePnPEvent : 0xfffff800`5f4b1e28 _NET_DEVICE_PNP_EVENT
   +0x018 NetPnPEventNotification : 0xfffff800`5f4b1e28 _NET_PNP_EVENT_NOTIFICATION
   +0x018 StatusIndication : 0xfffff800`5f4b1e28 _NDIS_STATUS_INDICATION
   +0x018 NextHandler      : 0xfffff800`5f4b1e28     void  +0
   +0x020 Nbls             : 0xffffc082`3655b9c0  _NET_BUFFER_LIST
   +0x028 Port             : 0
   +0x02c PathType         : 1 ( NdisSendCompletePath )
   +0x030 NumNbls          : 0
   +0x034 Flags            : 0
```

2，结合 TMS 反馈的分析结果（见 8/29 分析报告），ndis!ndisCallSendCompleteHandler 的完成过程中，需要调用 vwifimf!FilterSendNetBufferListsComplete

```
STACK_TEXT:
ffff8401`32bdd808 fffff801`40745107 : 00000000`0000001e ffffffff`80000003 fffff801`404e6235 ffff8401`32bde868 : nt!KeBugCheckEx
ffff8401`32bdd810 fffff801`406841b6 : fffff801`404e6235 fffff801`404e6218 ffff8401`32bdeaa0 ffff8401`32bde0b0 : nt!KiFatalFilter+0x1f
ffff8401`32bdd850 fffff801`4064554f : ffff8401`00000002 ffff8401`32bdfdf0 ffff8401`32bda000 ffff8401`32be1000 : nt!KeExpandKernelStackAndCalloutInternal$filt$0+0x16
ffff8401`32bdd890 fffff801`40673b5f : ffff8401`32bdfdf0 ffff8401`32bdde70 ffff8401`32bddf50 00000000`00000000 : nt!_C_specific_handler+0x9f
ffff8401`32bdd900 fffff801`405cc450 : ffff8401`32bddf50 00000000`00000000 ffff8401`32bdde70 fffff801`43c066d8 : nt!RtlpExecuteHandlerForException+0xf
ffff8401`32bdd930 fffff801`404d9c24 : ffff8401`32bde868 ffff8401`32bde5b0 ffff8401`32bde868 00000032`00360035 : nt!RtlDispatchException+0x430
ffff8401`32bde080 fffff801`4067c9c2 : ffffda8a`c5adf810 fffff801`47f7d880 00000000`00000000 fffff801`47c36915 : nt!KiDispatchException+0x144
ffff8401`32bde730 fffff801`40676681 : ffffda8a`cd331740 fffff801`4761cf2e 00000000`00000001 00000000`00000000 : nt!KiExceptionDispatch+0xc2
ffff8401`32bde910 fffff801`404e6236 : 00000000`00000000 00000000`00000000 00000000`00000001 fffff801`44ecb290 fffff801`44d0c0f8 : nt!KiBreakpointTrap+0x301
ffff8401`32bdeaa0 fffff801`406454eb : ffffda8a`d48afc30 ffff8401`32bdfdf0 fffff801`43c13588 ffff8401`32bdf0b8 : nt!KeCheckStackAndTargetAddress+0x46
ffff8401`32bdead0 fffff801`40673b5f : ffff8401`32bdfdf0 ffff8401`32bdf0b0 ffff8401`32bdf190 00000000`0010001f : nt!_C_specific_handler+0x3b
ffff8401`32bdeb40 fffff801`405cc450 : ffff8401`32bdf190 00000000`00000000 ffff8401`32bdf0b0 fffff801`43c066d8 : nt!RtlpExecuteHandlerForException+0xf
ffff8401`32bdeb70 fffff801`404d9c24 : ffff8401`32bdfaa8 ffff8401`32bdf7f0 ffff8401`32bdfaa8 ffffda8a`cdf28a30 : nt!RtlDispatchException+0x430
ffff8401`32bdf2c0 fffff801`4067c9c2 : 00000000`00001000 ffff8401`32bdfb50 ffff8000`00000000 00000032`00360035 : nt!KiDispatchException+0x144
ffff8401`32bdf970 fffff801`40678cae : ffffda8a`c4b149e0 ffffda8a`c4b14660 00000000`00000000 fffff801`405979ed : nt!KiExceptionDispatch+0xc2
ffff8401`32bdfb50 00000032`00360035 : fffff801`3ed59513 00000000`00000000 ffffda8a`d4c1fcb0 00000000`00000000 00000000`00000000 : nt!KiPageFault+0x42e
ffff8401`32bdfce8 fffff801`3ed59513 : 00000000`00000000 ffffda8a`d4c1fcb0 00000000`00000000 00000000`00000000 00000000`00000000 : 0x00000032`00360035
ffff8401`32bdfcf0 fffff801`3ed5d66d : ffffda8a`d4ae9640 00000000`00000000 ffff8401`32bdff10 ffffda8a`c4ad2660 : nwifi!Dot11SendCompletion+0x4b
ffff8401`32bdfd30 fffff801`43b766a3 : fffff801`43b78490 ffff8401`32bdfe20 00000000`00000000 fffff801`40597b21 : nwifi!Pt6SendComplete+0x1d
ffff8401`32bdfd60 fffff801`43b784ce : 00000000`00000200 00000000`00000000 ffff8401`32bdfe39 fffff801`4080106d : ndis!ndisCallSendCompleteHandler+0x33
ffff8401`32bdfda0 fffff801`40597a78 : 00000000`00000000 ffffda8a`c4ad1660 00000000`00000002 fffff801`47be30ea : ndis!ndisDataPathExpandStackCallback+0x3e
ffff8401`32bdfdf0 fffff801`405979ed : fffff801`43b78490 ffff8401`32bdff10 ffffda8a`cc9c2650 ffffda8a`cc07b028 : nt!KeExpandKernelStackAndCalloutInternal+0x78
ffff8401`32bdfe60 fffff801`43b9e104 : ffffda8a`c4abd660 ffffda8a`c4abd660 fffff801`00000000 ffffda8a`00000002 : nt!KeExpandKernelStackAndCalloutEx+0x1d
ffff8401`32bdfea0 fffff801`44641f92 : ffffda8a`c4ac3a01 ffffda8a`d305f030 ffff8401`00000000 00000000`00000000 : ndis!NdisFSendNetBufferListsComplete+0x28dc4
ffff8401`32bdff90 fffff801`43b766a3 : fffff801`44641e28 00000000`00000000 ffff8401`32be0190 fffff801`40597b21 : vwifimf!FilterSendNetBufferListsComplete+0x16a [d:\6.5\i
ffff8401`32bdffe0 fffff801`43b784ce : 00000000`000000ff ffffda8a`cd68e030 ffffda8a`cc07b010 ffffda8a`c4abd660 : ndis!ndisCallSendCompleteHandler+0x33
ffff8401`32be0020 fffff801`40597a78 : 00000000`00000000 ffffda8a`c4ad2660 ffffda8a`c4402100 00000000`00000000 : ndis!ndisDataPathExpandStackCallback+0x3e
ffff8401`32be0070 fffff801`405979ed : fffff801`43b78490 ffff8401`32be0190 ffffda8a`cc9c2950 ffffda8a`d4e88ba0 : nt!KeExpandKernelStackAndCalloutInternal+0x78
ffff8401`32be00e0 fffff801`43b9e104 : 00000000`00000103 00000000`00000000 ffffda8a`cd68e030 00000002`00000000 : nt!KeExpandKernelStackAndCalloutEx+0x1d
```

3, 是否需要调用 vwifimf!FilterSendNetBufferListsComplete 的依据为判断传入参数-NBL 数据结构中的 SourceHandle 和 NDIS_HANDLE 中的 FilterHandle 是否相同，即是否为经重组的 802.1x 数据包。

基于上述流程分析，现在的问题集中在满足什么条件的数据包需要重组，由于 vwifimf.sys 作为过滤驱动在调用初始阶段会申请一段内存地址 pool，并在之后的调用过程中进行单独维护，因此 vwifimf.sys 决定以下几个因素：

- filter 的条件由 vwifimf.sys 声明
- 重组的数据包场景由 vwifimf.sys 定义
- 重组数据包之后的调用流程由 vwifimf.sys 决定

以此次收集的 dump 日志为例，当前需要处理的 NBL 为 0xffffc082`3655b9c0 。通过 trace NBL，发现最初的 Source 指向 Native WiFi Filter Driver，以表明这并不是一个由 vwifimf.sys 分配出来的 NBL.。并且经验证，Source 的 NBL 中，nwifi.sys 的相关结果是有效的。

```
kd> !nbl 0xffffc082`3655b9c0
    NBL             ffffc0823655b9c0    Next NBL          NULL
    First NB        ffffc0822345db10    Source            ffffc08217eee8a0 - Native WiFi Filter
Driver-0000
    Context stack   ffffc0823655bb40    Pool              ffffc08220a32640 - nwifi
    Flags           FREED_BACK_TO_NDIS
```

但是当 NBL 经过 vwifimf.sys 之后，传入 NdisFSendNetBufferListsComplete 的 NBL 变成另外一个由 vwifimf.sys 分配的 NBL，即 vwifimf.sys 完成了对该数据包的重组。

```
kd> !ndiskd.nbl 0xffffc082`25917500
    NBL             ffffc08225917500    Next NBL          NULL
    First NB        ffffc08225917680    Source            ffffc082207db250 - NDIS Sample
LightWeight Filter 1-0000
    Context stack   ffffc0822b503210    Pool              ffffc0821f1c49c0 -
    Flags           NBL_ALLOCATED, NBL_CONTEXT_ALLOCATED
```

这时 NBL 的 Source Handle 指向 Filter Handle ffffc082207db250，这个 NBL 对于 nwifi.sys 来说是非法的，vwifimf.sys 不应该再调用 ndis!NdisFSendNetBufferListsComplete，因此导致系统蓝屏。具体可参见以下文档：

https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ndis/nf-ndis-ndisfsendnetbufferlistscomplete

Note A filter driver should keep track of send requests that it originates and make sure that it does not call the **NdisFSendNetBufferListsComplete** function when such requests are complete.

问题总结:

==========

请 TMS 厂商根据自身代码的条件声明，场景定义进行对于 802.1x 数据包重组部分的代码修改。
如确实已完成对 802.1x 数据包的重组行为，则需要加强代码验证逻辑，在验证重组数据包的安全性之后，传入 Windows 允许的 handle 信息，或跳过调用 ndis!NdisFSendNetBufferListsComplete 的代码部分。

经与用户沟通，目前完成此 dump 的分析工作，同意关闭此 case。

以上，如您后续有任何问题，可随时与我们联系，谢谢

李琦 Li Qi
神州网信技术有限公司
C&M Information Technologies Co.,Ltd.
服务电话： 4008180055
电子邮箱 Email: liqi@cmgos.com

发件人: Li Qi <liqi@cmgos.com>
发送时间: 2020 年 9 月 4 日 13:43
收件人: 吴毓杰 <win10sup@sdc.icbc.com.cn>
抄送: Li Qi <liqi@cmgos.com>
主题: [案例号: CAS-02903-W1P8K5] % |P3|ICBC|神州网信政府版系统在使用无线网络过程蓝屏 % 初次响应 CMIT:0001958

吴毓杰 先生/女士，您好!

感谢您联系神州网信技术支持中心。 我是技术支持工程师 李琦 。 很高兴能有机会协助您解决该问题。 您可随时通过邮件回复以及该问题事件号码 CAS-02903-W1P8K5 与我联系。

如果您有任何其他疑问，请随时与我联系。

此致，

敬礼
