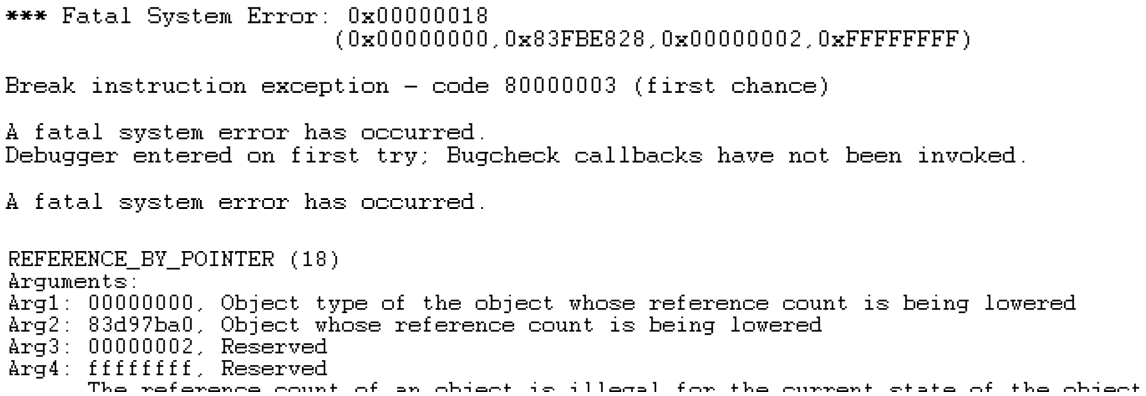
firozimaysam@gmail.com   @R00tkitSmm

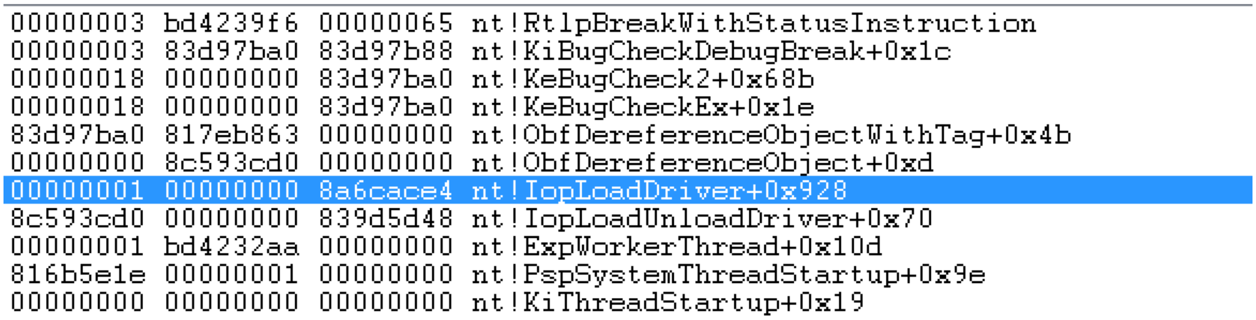
Double IoDereferenceObject call Bug in UnThreat AV Driver:
http://www.unthreat.com/

when i start to find vulnerability in UnThreat AV i select sbapifsl.sys but this driver is not loaded in default os config   i start to load driver for live debugging after i start Driver i get BSOD with bug check code REFERENCE_BY_POINTER

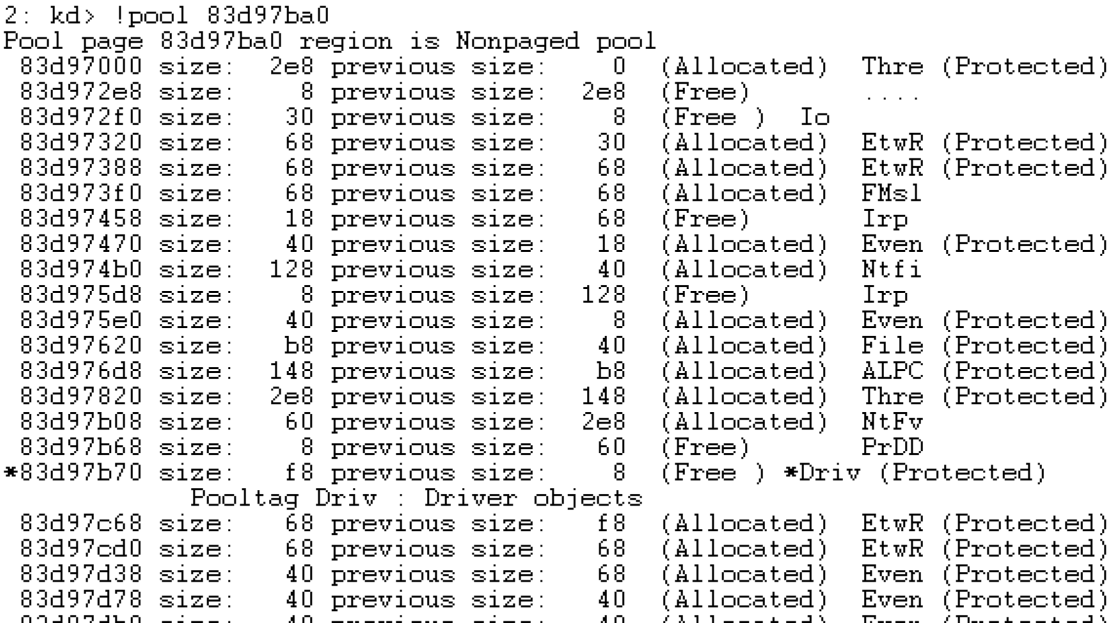


```
*** Fatal System Error: 0x00000018
                       (0x00000000,0x83FBE828,0x00000002,0xFFFFFFFF)

Break instruction exception - code 80000003 (first chance)

A fatal system error has occurred.
Debugger entered on first try; Bugcheck callbacks have not been invoked.

A fatal system error has occurred.


REFERENCE_BY_POINTER (18)
Arguments:
Arg1: 00000000, Object type of the object whose reference count is being lowered
Arg2: 83d97ba0, Object whose reference count is being lowered
Arg3: 00000002, Reserved
Arg4: ffffffff, Reserved
      The reference count of an object is illegal for the current state of the object
```

i checked Call Stack and with help of Arg2 (0x83d97ba0) and windbg it revealed that IppLoadDriver call ObDereferenceObject with pointer to free object



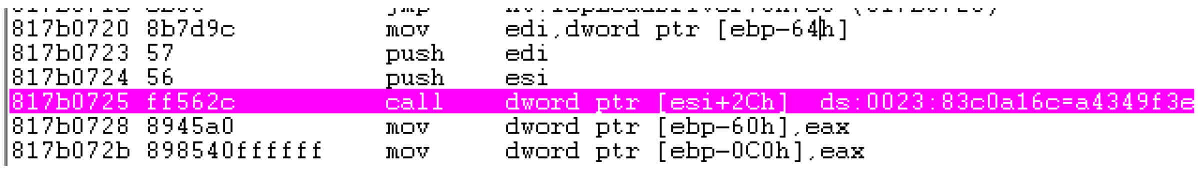
```
00000003 bd4239f6 00000065 nt!RtlpBreakWithStatusInstruction
00000003 83d97ba0 83d97b88 nt!KiBugCheckDebugBreak+0x1c
00000018 00000000 83d97ba0 nt!KeBugCheck2+0x68b
00000018 00000000 83d97ba0 nt!KeBugCheckEx+0x1e
83d97ba0 817eb863 00000000 nt!ObfDereferenceObjectWithTag+0x4b
00000000 8c593cd0 00000000 nt!ObfDereferenceObject+0xd
00000001 00000000 8a6cace4 nt!IopLoadDriver+0x928
8c593cd0 00000000 839d5d48 nt!IopLoadUnloadDriver+0x70
00000001 bd4232aa 00000000 nt!ExpWorkerThread+0x10d
816b5e1e 00000001 00000000 nt!PspSystemThreadStartup+0x9e
00000000 00000000 00000000 nt!KiThreadStartup+0x19
```



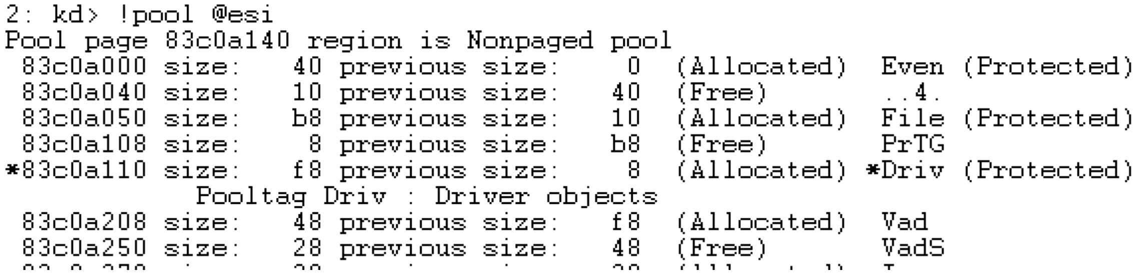
```
2: kd> !pool 83d97ba0
Pool page 83d97ba0 region is Nonpaged pool
 83d97000 size:  2e8 previous size:    0  (Allocated)  Thre (Protected)
 83d972e8 size:    8 previous size:  2e8  (Free)       ....
 83d972f0 size:   30 previous size:    8  (Free )  Io
 83d97320 size:   68 previous size:   30  (Allocated)  EtwR (Protected)
 83d97388 size:   68 previous size:   68  (Allocated)  EtwR (Protected)
 83d973f0 size:   68 previous size:   68  (Allocated)  FMsl
 83d97458 size:   18 previous size:   68  (Free)       Irp
 83d97470 size:   40 previous size:   18  (Allocated)  Even (Protected)
 83d974b0 size:  128 previous size:   40  (Allocated)  Ntfi
 83d975d8 size:    8 previous size:  128  (Free)       Irp
 83d975e0 size:   40 previous size:    8  (Allocated)  Even (Protected)
 83d97620 size:   b8 previous size:   40  (Allocated)  File (Protected)
 83d976d8 size:  148 previous size:   b8  (Allocated)  ALPC (Protected)
 83d97820 size:  2e8 previous size:  148  (Allocated)  Thre (Protected)
 83d97b08 size:   60 previous size:  2e8  (Allocated)  NtFv
 83d97b68 size:    8 previous size:   60  (Free)       PrDD
*83d97b70 size:   f8 previous size:    8  (Free ) *Driv (Protected)
            Pooltag Driv : Driver objects
 83d97c68 size:   68 previous size:   f8  (Allocated)  EtwR (Protected)
 83d97cd0 size:   68 previous size:   68  (Allocated)  EtwR (Protected)
 83d97d38 size:   40 previous size:   68  (Allocated)  Even (Protected)
 83d97d78 size:   40 previous size:   40  (Allocated)  Even (Protected)
 83d97db0 size:   40 previous size:   40  (Allocated)  Even (Protected)
```

i start debugging and find out this object is created by nt!ObCreateObject in offset 1A33FF of Kernel image this is Driver Object that is created when os try to load sys file
and after init this object kernel will call DriverEntry and pass inited Driver object to it

and i set breakpoint before and after DriverEntry to check Object state (Reference count and Object Pool memory)

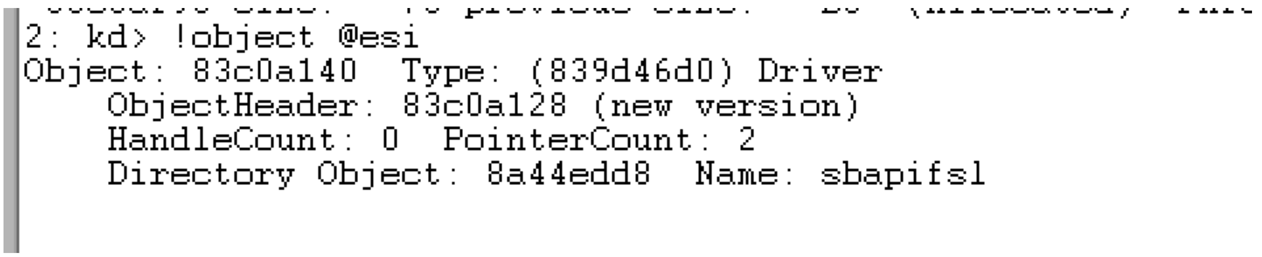


```
817b0720 8b7d9c          mov     edi,dword ptr [ebp-64h]
817b0723 57              push    edi
817b0724 56              push    esi
817b0725 ff562c          call    dword ptr [esi+2Ch]  ds:0023:83c0a16c=a4349f3e
817b0728 8945a0          mov     dword ptr [ebp-60h],eax
817b072b 898540ffffff    mov     dword ptr [ebp-0C0h],eax
```



```
2: kd> !pool @esi
Pool page 83c0a140 region is Nonpaged pool
 83c0a000 size:   40 previous size:    0  (Allocated)  Even (Protected)
 83c0a040 size:   10 previous size:   40  (Free)        ..4.
 83c0a050 size:   b8 previous size:   10  (Allocated)  File (Protected)
 83c0a108 size:    8 previous size:   b8  (Free)       PrTG
*83c0a110 size:   f8 previous size:    8  (Allocated) *Driv (Protected)
            Pooltag Driv : Driver objects
 83c0a208 size:   48 previous size:   f8  (Allocated)  Vad
 83c0a250 size:   28 previous size:   48  (Free)       VadS
```

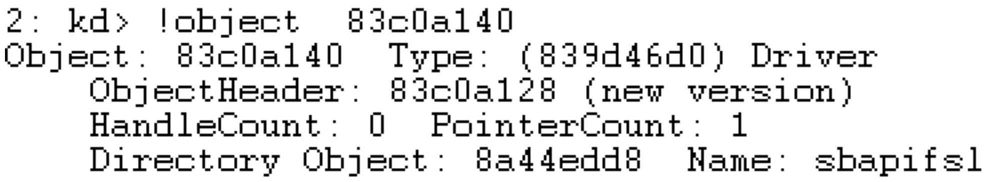
!pool @ebp



```
2: kd> !object @esi
Object: 83c0a140  Type: (839d46d0) Driver
    ObjectHeader: 83c0a128 (new version)
    HandleCount: 0  PointerCount: 2
    Directory Object: 8a44edd8  Name: sbapifsl
```

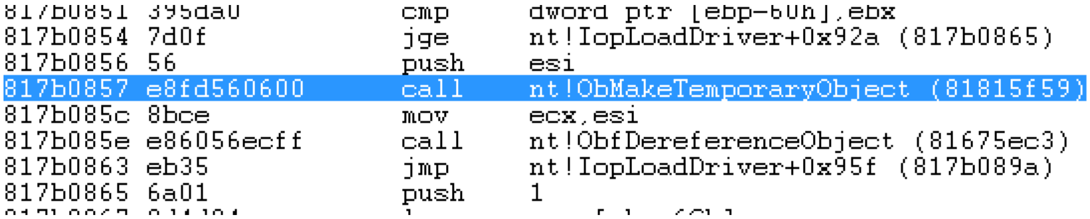
object is driver and with pointer Count is 2

after DriverEntry We have PointerCount 1 , i don't think this is logical and reference count must be same after and before DriverEntry (i checked it with another Driver)
so it was absolutely sbapifsl.sys bug it Dereference object for one extra time



```
2: kd> !object  83c0a140
Object: 83c0a140  Type: (839d46d0) Driver
    ObjectHeader: 83c0a128 (new version)
    HandleCount: 0  PointerCount: 1
    Directory Object: 8a44edd8  Name: sbapifsl
```

after DriverEntry return , code path lead to ObMakeTemporaryObject and inside this function there was another ObfDereferenceObject and this will cause os free Object because it Pointercount reach to Zero , after ObMakeTemporaryObject return we have another ObfDereferenceObject and this object will use free pool and object



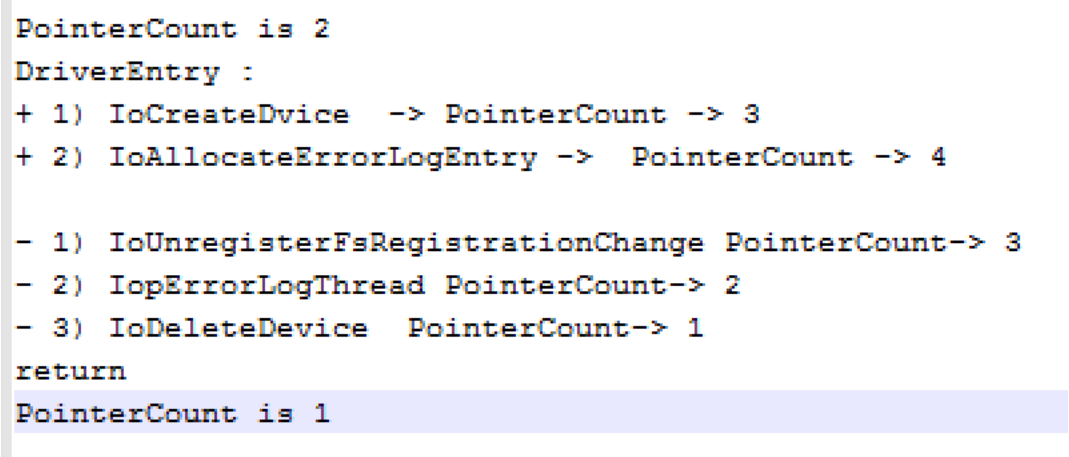
```
817b0851 395da0          cmp     dword ptr [ebp-60h],ebx
817b0854 7d0f            jge     nt!IopLoadDriver+0x92a (817b0865)
817b0856 56              push    esi
817b0857 e8fd560600      call    nt!ObMakeTemporaryObject (81815f59)
817b085c 8bce            mov     ecx,esi
817b085e e86056ecff      call    nt!ObfDereferenceObject (81675ec3)
817b0863 eb35            jmp     nt!IopLoadDriver+0x95f (817b089a)
817b0865 6a01            push    1
```

offset 1A3857 ->>> ObfDereferenceObject —> BSOD

now We must find why driver use ObfDereferenceObject one more extra time
i set write breakpoint in OBJECT_HEADER and let os call DriverEntry

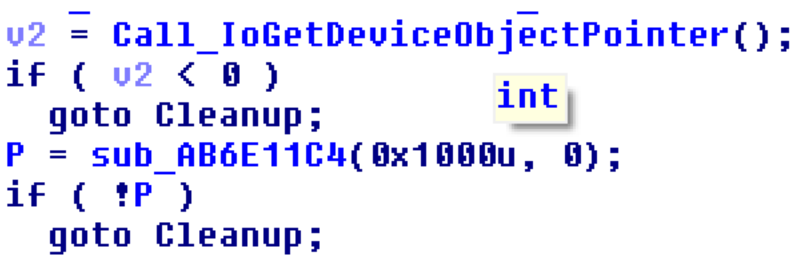
kd> ba w 4 84a7b668

so this breakpoint will trigger for every increment and decrement   PointerCount in OBJECT_HEADER i find out there was tow ObfReferenceObject call but tree ObfDereferenceObject and it was related to IoUnregisterFsRegistrationChange



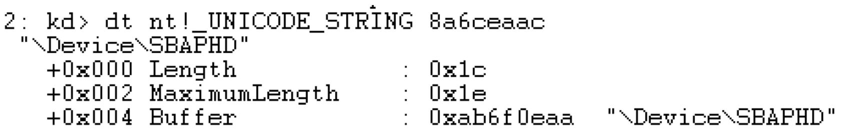
```
PointerCount is 2
DriverEntry :
+ 1) IoCreateDvice   -> PointerCount -> 3
+ 2) IoAllocateErrorLogEntry ->  PointerCount -> 4

- 1) IoUnregisterFsRegistrationChange PointerCount-> 3
- 2) IopErrorLogThread PointerCount-> 2
- 3) IoDeleteDevice  PointerCount-> 1
return
PointerCount is 1
```

so Driver Call IoUnregisterFsRegistrationChange without first call IoRegisterFsRegistrationChange if we check DriverEntry it call some function and if one of them fails it will jump to clean up without call IoRegisterFsRegistrationChange and in cleanup we have IoUnregisterFsRegistrationChange



```
v2 = Call_IoGetDeviceObjectPointer();
if ( v2 < 0 )
  goto Cleanup;
P = sub_AB6E11C4(0x1000u, 0);
if ( !P )
  goto Cleanup;
```

Call_IoGetDeviceObjectPointer function use \Device\SBAPHD as UNICODE_STRING but in default AV installation we dot have this device so IoGetDeviceObjectPointer fail and return error :(



```
2: kd> dt nt!_UNICODE_STRING 8a6ceaac
 "\Device\SBAPHD"
   +0x000 Length           : 0x1c
   +0x002 MaximumLength    : 0x1e
   +0x004 Buffer           : 0xab6f0eaa  "\Device\SBAPHD"
```

first time i think about spray Pool with same size object before free and reuse and with this method i will force this bug free my object and again i spray pool with another Object with same size and force OS to type confusion   but between free and reuse we don't have any time for spray pool !!!! if there was any idea how we can exploit this type of bug let me know