

# Needle in A Haystack: Catch Multiple Zero-days Using Sandbox

**Qi Li**

Security Development Engineer

**Quan Jin**

Vulnerability Mining and Exploiting Engineer

2019-5-30

# About Us



**Qi Li** (@leeqwind)

**360 Core Security Advanced Threat  
Automation Team**

**Security Development Engineer**



**Quan Jin** (@jq0904)

**360 Core Security Advanced Threat  
Automation Team**

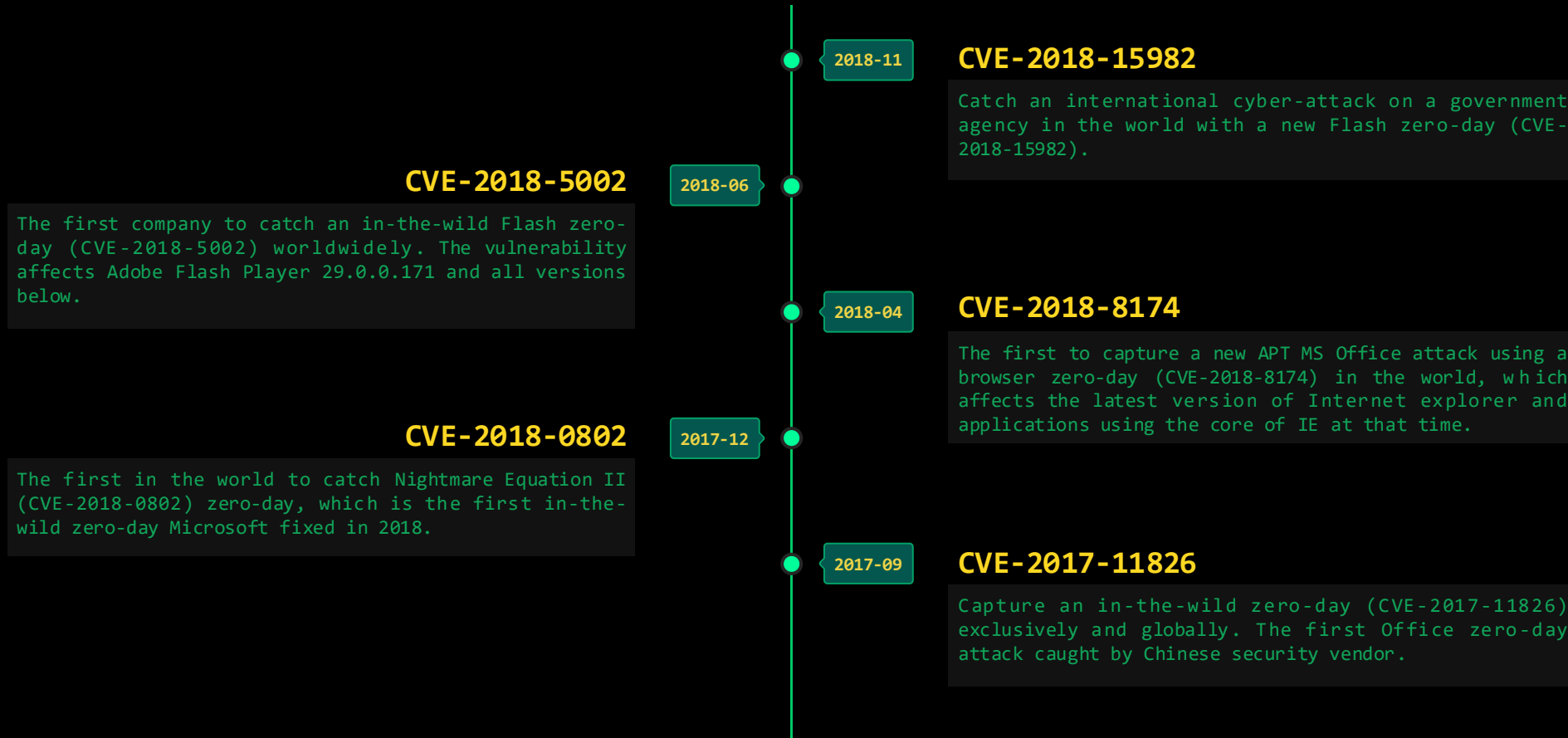
**Vulnerability Mining and Exploiting Engineer**

# Outline



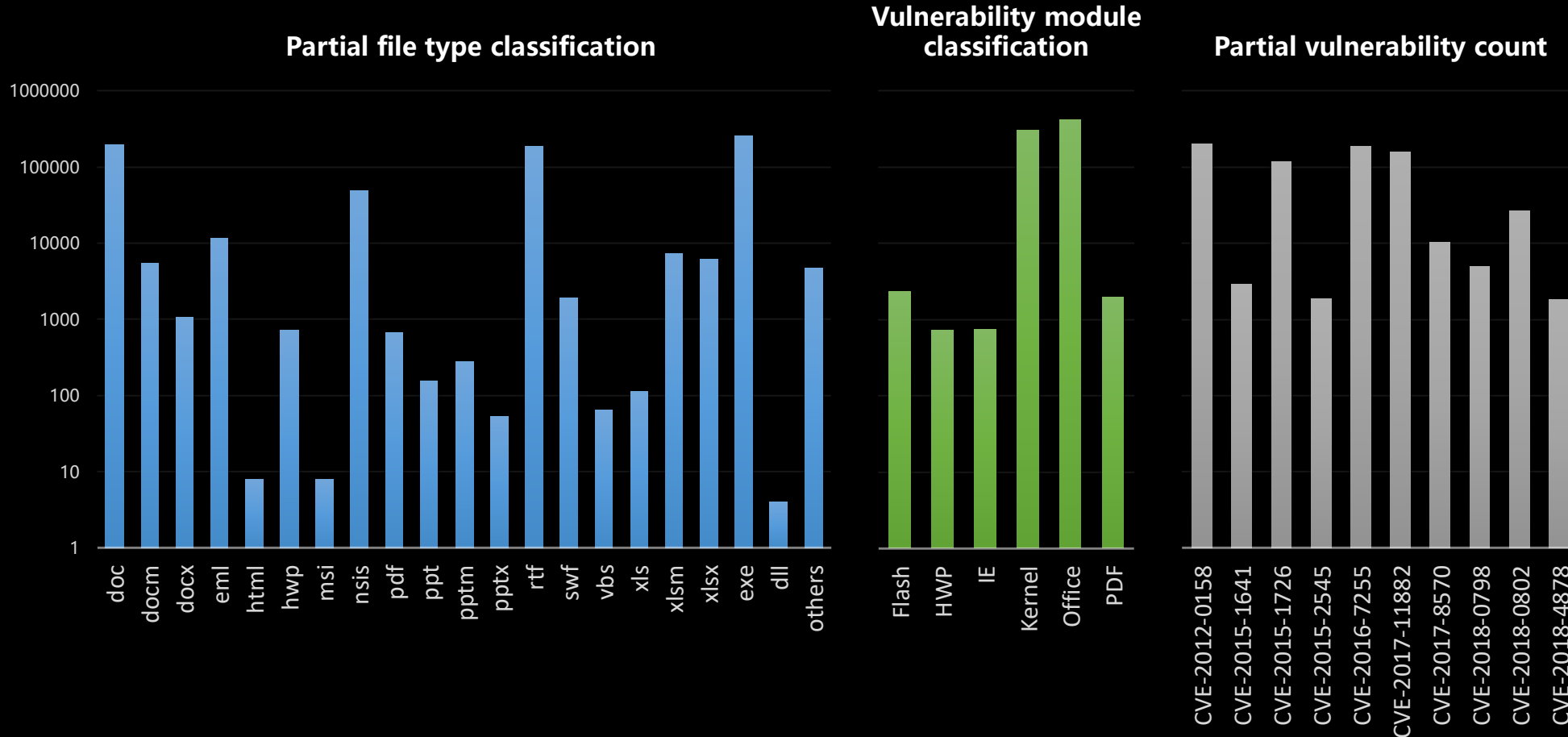
- **Advanced Threat Automation and Sandbox**
- **Find in-the-wild zero-days using Sandbox**

# Cyber Attacks are Everywhere



The five in-the-wild exploiting zero-day attacks we captured up to now

# Cyber Attacks are Everywhere

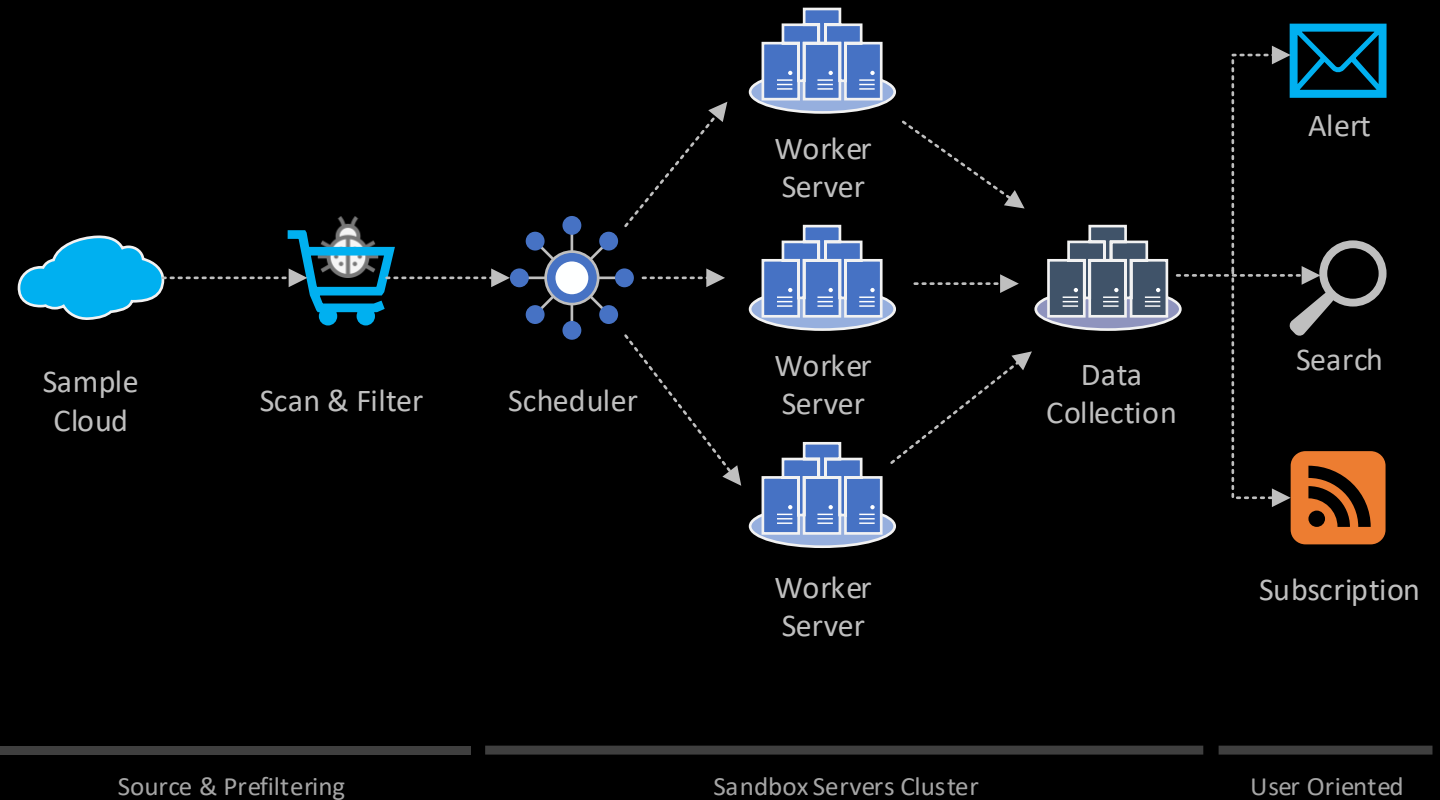


Statistics for some of the N-day exploits we detected from March 2018 to March 2019

# Advanced Threat Automation



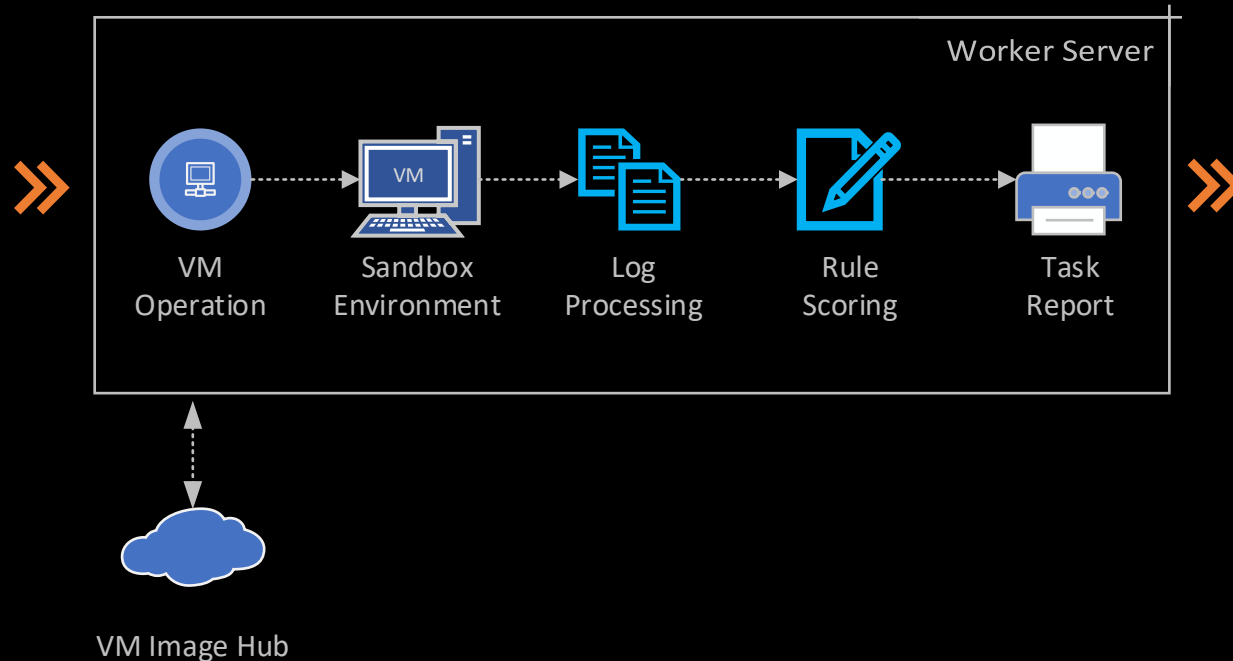
- **Large-scale Sample Cloud**
- **Static Anti-virus Engine**
  - AVE QEX QVM
- **Sample Pre-filtering Strategy**
- **Sandbox Servers Cluster**
  - Virtual Machine Isolation Environments
  - Sandbox Detection Engine
  - Rule Scoring System
- **Result Alarm and Response**



# Advanced Threat Automation



- **Large-scale Sample Cloud**
- **Static Anti-virus Engine**
  - AVE QEX QVM
- **Sample Pre-filtering Strategy**
- **Sandbox Servers Cluster**
  - Virtual Machine Isolation Environments
  - Sandbox Detection Engine
  - Rule Scoring System
- **Result Alarm and Response**



**Sandbox Detection Engine**

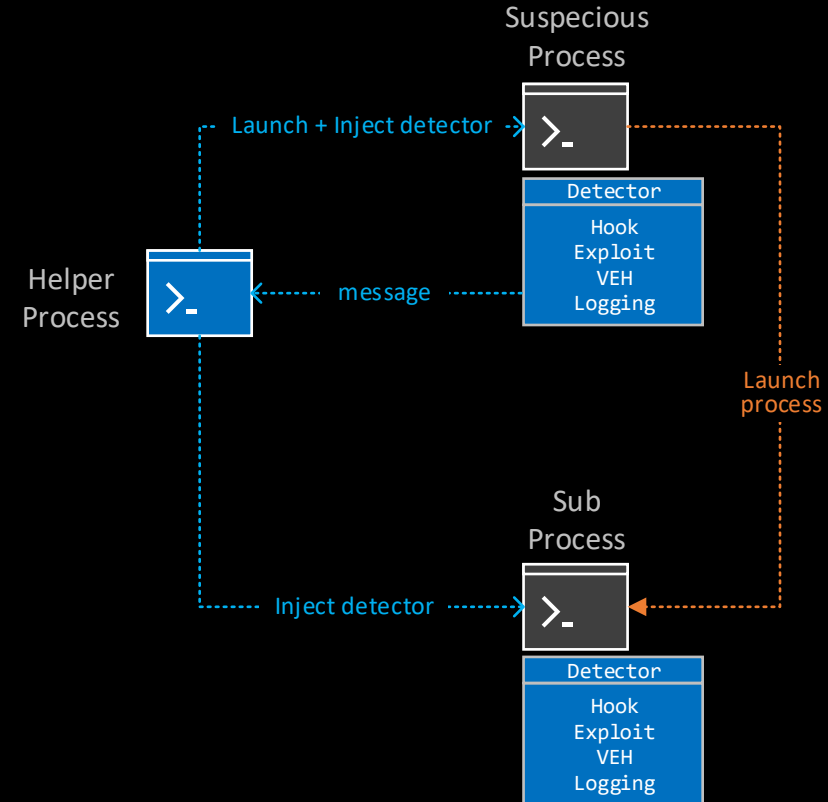
**How to do it?**



# Sandbox Detection Engine

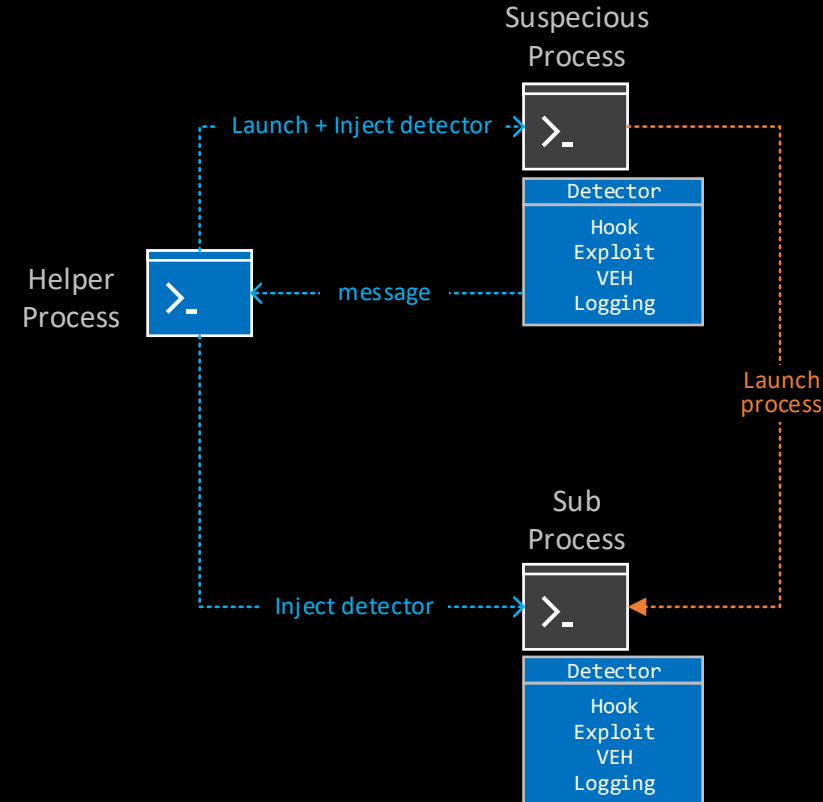


- **Initial Scenario: Dynamic Library**
- **Inject into target processes to work**
- **Hook export functions of system libs**



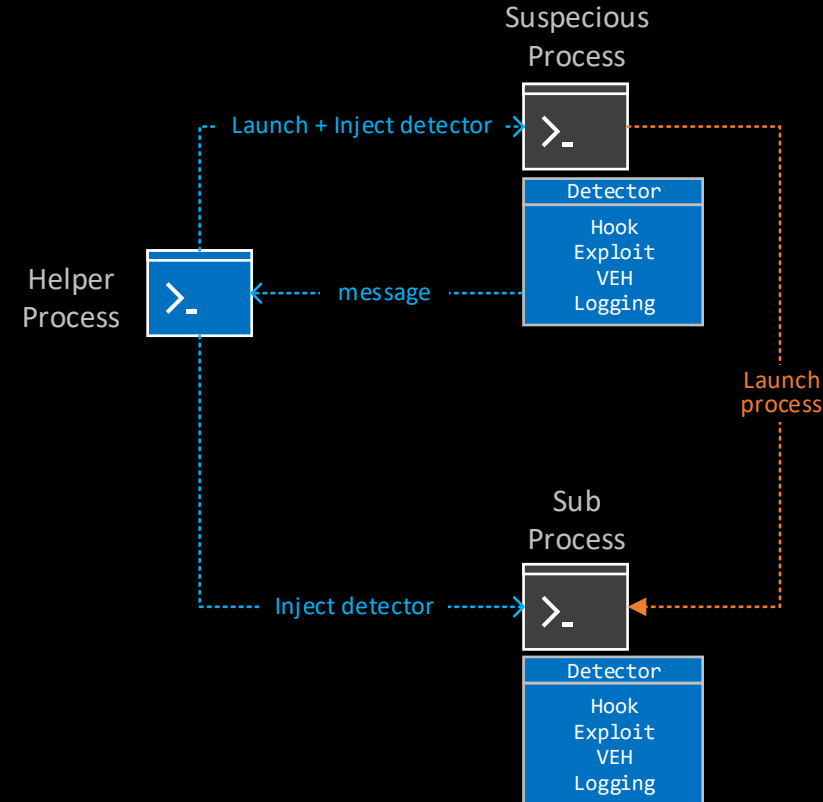
# Sandbox Detection Engine

- Initial Scenario: Dynamic Library
- Inject into target processes to work
- Hook export functions of system libs
- Lightweight 😊



# Sandbox Detection Engine

- Initial Scenario: Dynamic Library
- Inject into target processes to work
- Hook export functions of system libs
- Lightweight 😊
- **Is that enough?**



# Sandbox Detection Engine

- Initial Scenario: Dynamic Library
- Inject into target processes to work
- Hook export functions of system libs
- Lightweight 😊
- **Is that enough?**
- Can be detected easily 😞



Suspicious Process

Launch + inject detector

Name	Path	Description	Company Name
ATL90.dll	C:\Windows\winsxs\x86_microsoft.vc90.atl_7c8b71c...	ATL Module for Windows (U...	Microsoft Corporation
AudioSes.dll	C:\Windows\System32\AudioSes.dll	Audio Session	Microsoft Corporation
authui.dll	C:\Windows\System32\authui.dll	Windows 身份验证 UI	Microsoft Corporation
avrt.dll	C:\Windows\System32\avrt.dll	Multimedia Realtime Runtime	Microsoft Corporation
batmeter.dll	C:\Windows\System32\batmeter.dll	电池电量辅助程序 DLL	Microsoft Corporation
bthprops.cpl	C:\Windows\System32\bthprops.cpl	Bluetooth 控制项小程序	Microsoft Corporation
bthprops.cpl.mui	C:\Windows\System32\zh-CN\bthprops.cpl.mui	Bluetooth 控制项小程序	Microsoft Corporation
cfmgmgr32.dll	C:\Windows\System32\cfmgmgr32.dll	Configuration Manager	Microsoft Corporation
clbcatq.dll	C:\Windows\System32\clbcatq.dll	COM+ Configuration Catalog	Microsoft Corporation
comctl32.dll	C:\Windows\winsxs\x86_microsoft.windows.com...	用户体验控件库	Microsoft Corporation
comctl32.dll.mui	C:\Windows\winsxs\x86_microsoft.windows.c...-co...	用户体验控件库	Microsoft Corporation
credssp.dll	C:\Windows\System32\credssp.dll	Credential Delegation Securi...	Microsoft Corporation
crypt32.dll	C:\Windows\System32\crypt32.dll	加密 API32	Microsoft Corporation

Inject detector

Detector

- Hook
- Exploit
- VEH
- Logging

**Detectable Engine Module**

# Sandbox Detection Engine

- Initial Scenario: Dynamic Library
- Inject into target processes to work
- Hook export functions of system libs
- Lightweight 😊
- **Is that enough?**
- Can be detected easily 😞
- Can be bypassed easily 😞

```
00002960          sub_2960          proc near
00002960 49 89 C3          mov     r10, rcx
00002963 B8 7F 10 00 00    mov     eax, 107Fh
00002968 0F 05           syscall
0000296A C3             retn
0000296A          sub_2960          endp
```

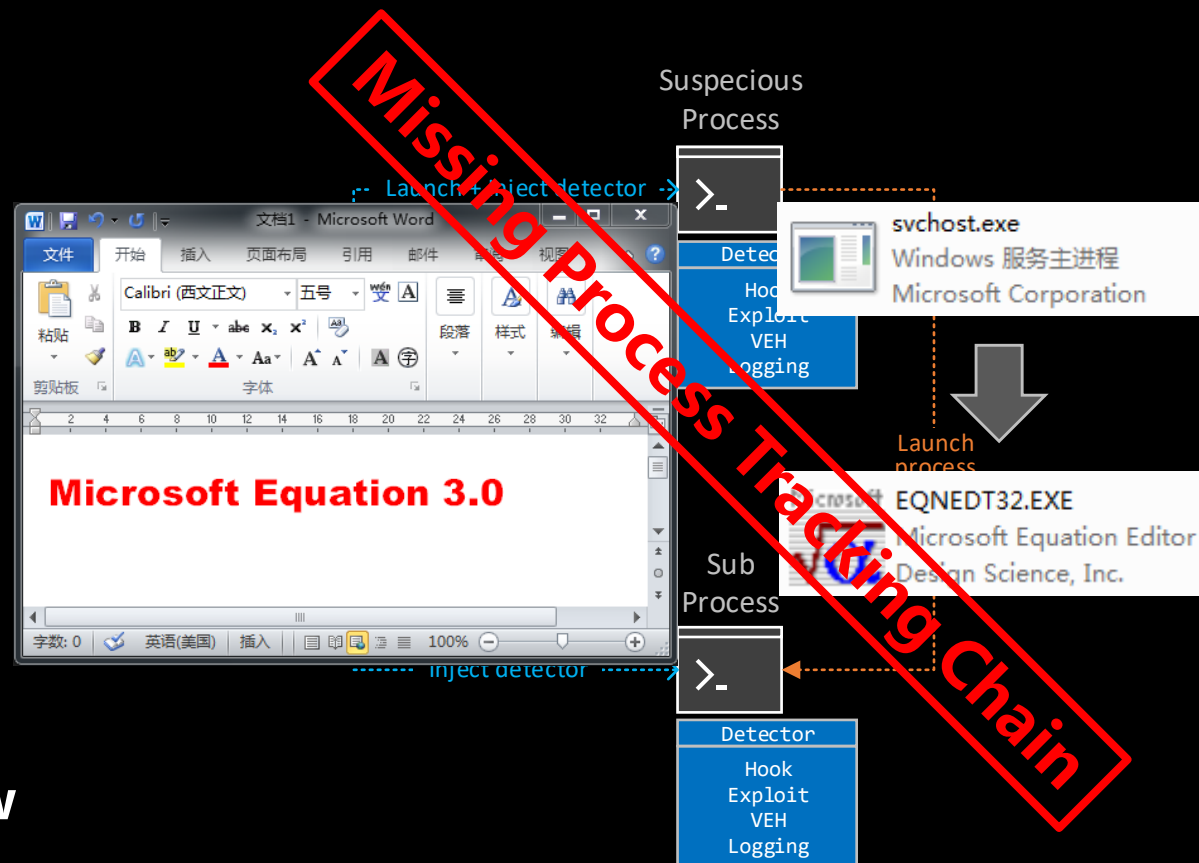
```
0:007> uf 077fd000
077fd000 81ec00080000 ... sub esp,800h
077fd006 60 ... pushad
[ ... ]
077fd01d ba1d5fc177 ... mov edx, offset ntdll!NtPrivilegedServiceAuditAlarm+0x5 (77ca5edd)
077fd022 8d45fc ... lea eax,[ebp-4]
077fd025 50 ... push eax
[ ... ]
077fd032 b8d7000000 ... mov eax,0D7h
077fd037 ffd2 ... call edx
[ ... ]
0:007> uf ntdll!NtPrivilegedServiceAuditAlarm
ntdll!NtPrivilegedServiceAuditAlarm:
77ca5ed8 b8d3000000 ... mov eax,0D3h
77ca5edd ba0003fe7f ... mov edx,offset SharedUserData!SystemCallStub (7ffe9300)
77ca5ee2 ff12 ... call dword ptr [edx]
77ca5ee4 c21400 ... ret 14h
0:000> uf ntdll!NtProtectVirtualMemory
ntdll!NtProtectVirtualMemory:
77ca5f18 b8d7000000 ... mov eax,0D7h
77ca5f1d ba0003fe7f ... mov edx,offset SharedUserData!SystemCallStub (7ffe9300)
77ca5f22 ff12 ... call dword ptr [edx]
77ca5f24 c21400 ... ret 14h
```

**Bypass User-mode Detection**

# Sandbox Detection Engine

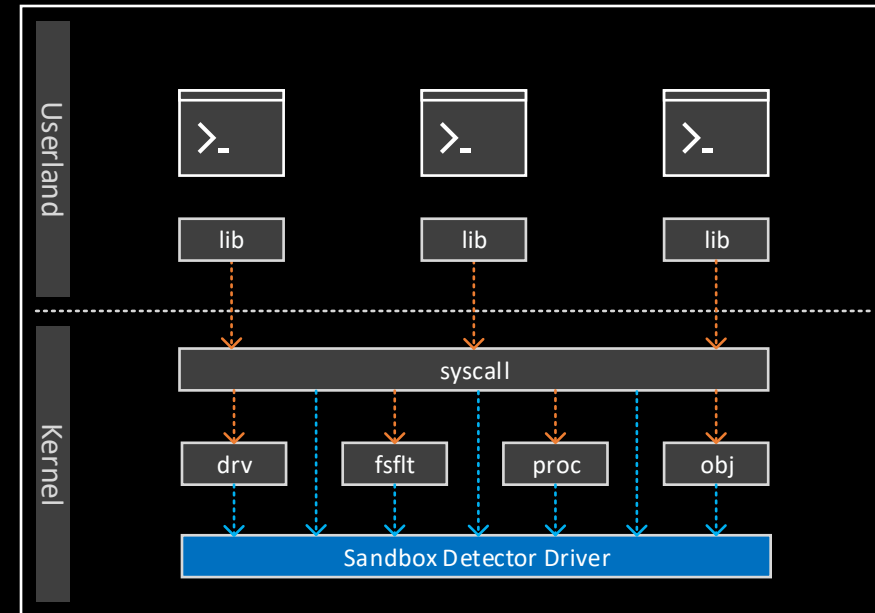


- Initial Scenario: Dynamic Library
- Inject into target processes to work
- Hook export functions of system libs
- Lightweight 😊
- **Is that enough?**
- Can be detected easily 😞
- Can be bypassed easily 😞
- Easy to lose the tracking chain to new processes launched remotely 😞



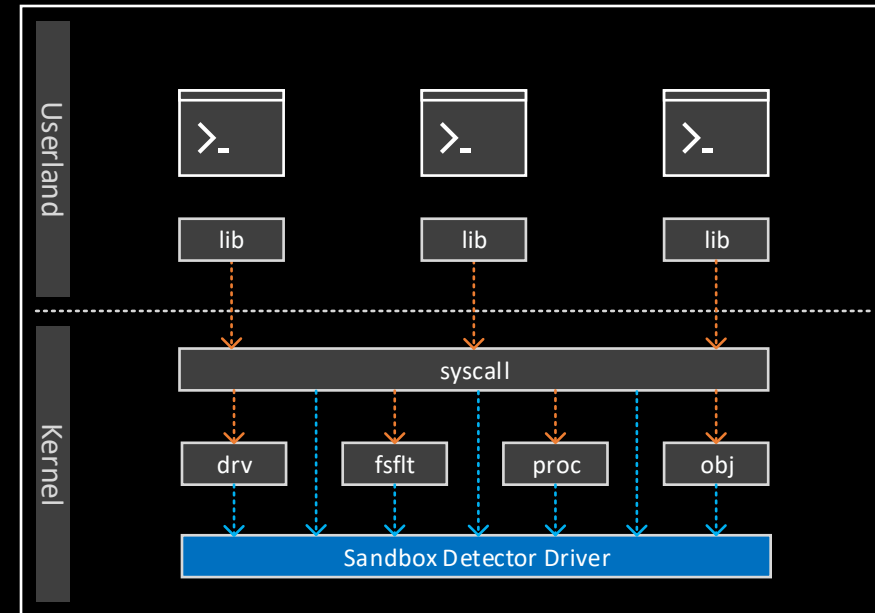
# Sandbox Detection Engine

- The 2<sup>nd</sup> Option: Driver
- Monitor system call from target in kernel
- System callbacks, notifications, filters



# Sandbox Detection Engine

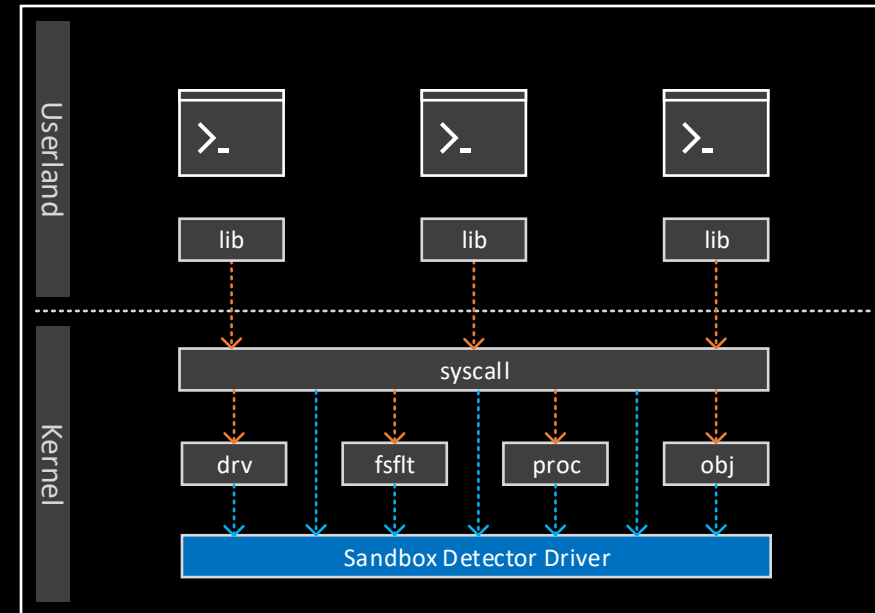
- The 2<sup>nd</sup> Option: Driver
- Monitor system call from target in kernel
- System callbacks, notifications, filters
- More complete monitoring coverage 🤖
- More comprehensive stain tracking 🤖





# Sandbox Detection Engine

- The 2<sup>nd</sup> Option: Driver
- Monitor system call from target in kernel
- System callbacks, notifications, filters
- More complete monitoring coverage 😊
- More comprehensive stain tracking 😊
- **Is that all right?**



# Sandbox Detection Engine



- The 2<sup>nd</sup> Option: Driver
- Monitor system call from target in kernel
- System callbacks, notifications, filters
- More complete monitoring coverage 😊
- More comprehensive stain tracking 😊
- Is that all right?
- PATCH GUARD for 64-bit OS 😞

```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

Modification of system code or a critical data structure was detected.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

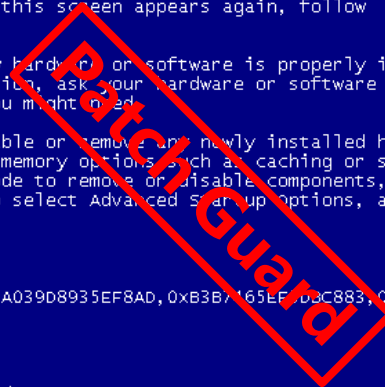
Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any windows updates you might need.

If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Start up options, and then
select safe Mode.

Technical information:

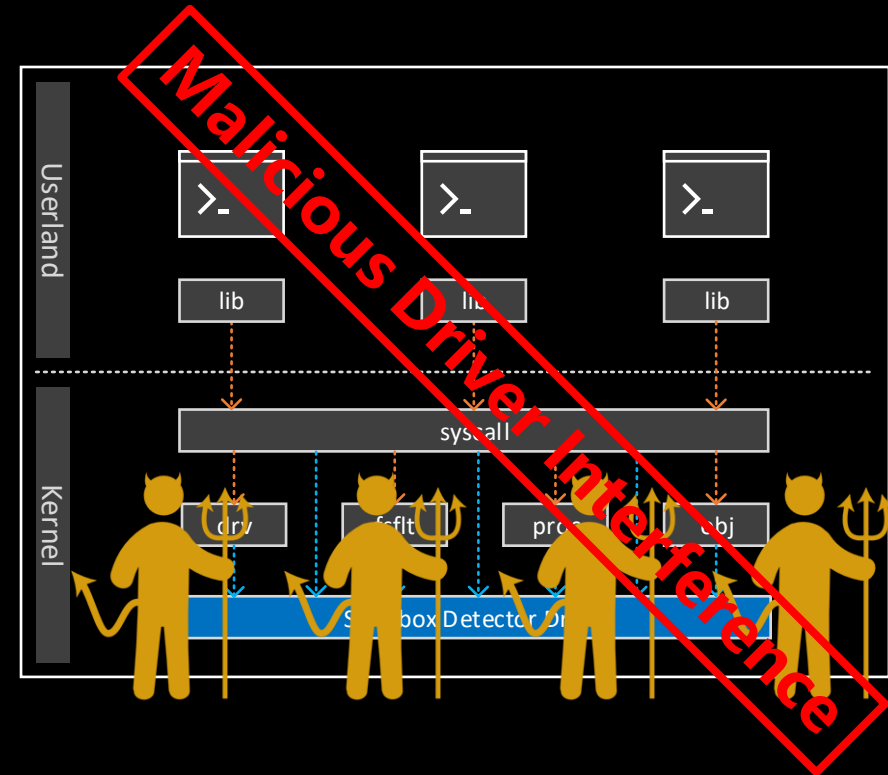
*** STOP: 0x00000109 (0xA3A039D8935EF8AD,0xB3B7165EF,0DC883,0xFFFFF80003E704D0,0
x000000000000000001)

Collecting data for crash dump ...
Initializing disk for crash dump ...
```



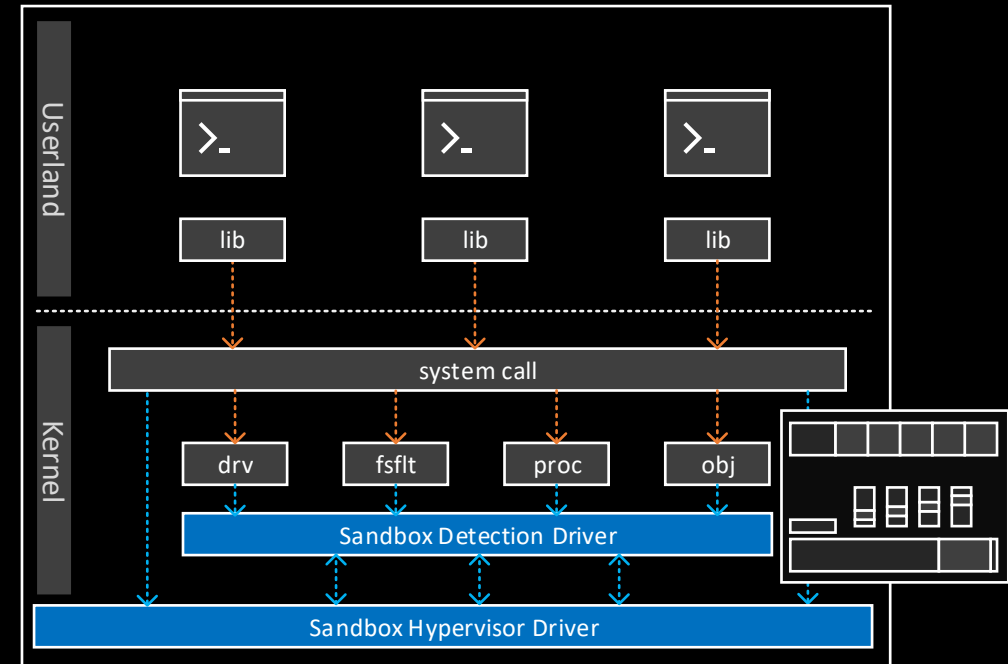
# Sandbox Detection Engine

- The 2<sup>nd</sup> Option: Driver
- Monitor system call from target in kernel
- System callbacks, notifications, filters
- More complete monitoring coverage 😊
- More comprehensive stain tracking 😊
- **Is that all right?**
- PATCH GUARD for 64-bit OS 😞
- Interference from malwares with drivers 😞



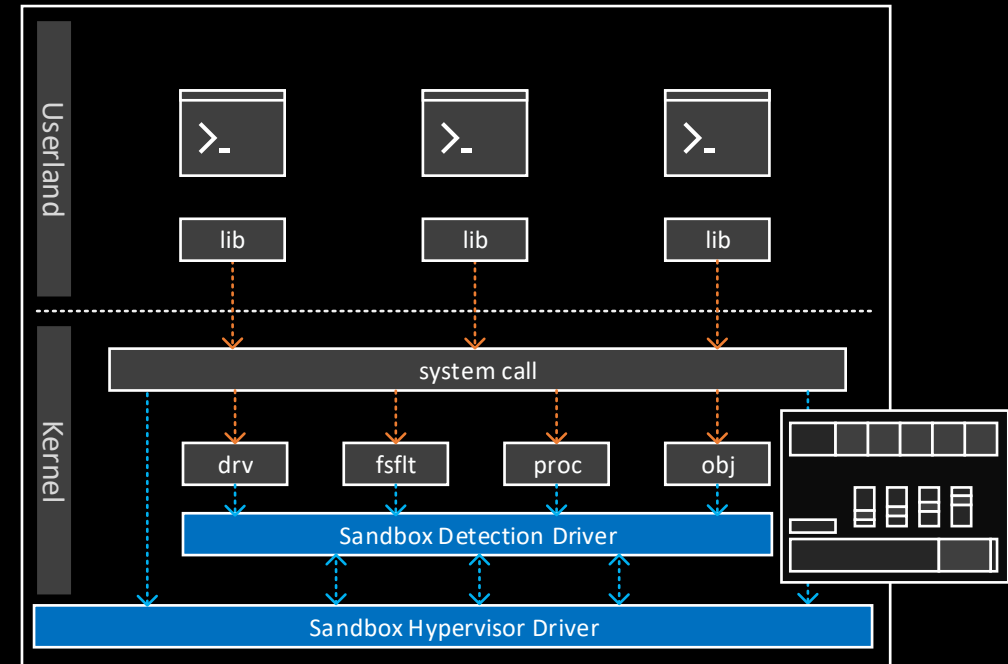
# Sandbox Detection Engine

- The 3<sup>rd</sup> Option: Virtualization-based Driver
- Virtualization-based system call monitoring
- R/W access to sensitive memory monitoring



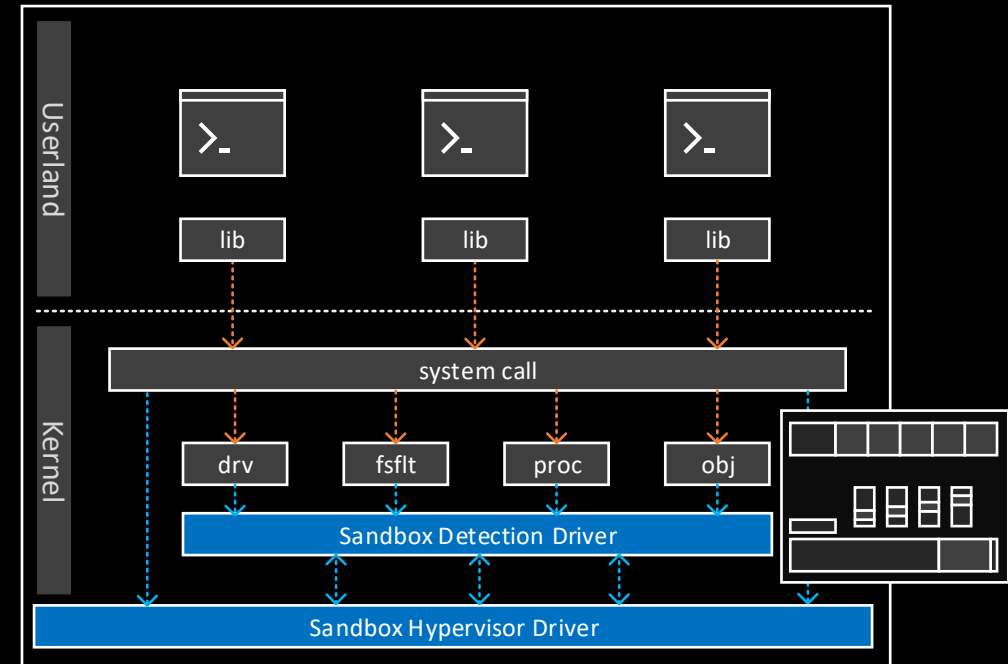
# Sandbox Detection Engine

- The 3<sup>rd</sup> Option: Virtualization-based Driver
- Virtualization-based system call monitoring
- R/W access to sensitive memory monitoring
- Avoid BSOD caused by PATCH GUARD 😊
- Protect private driver code and data 😊
- Expand more comprehensive detection 😊



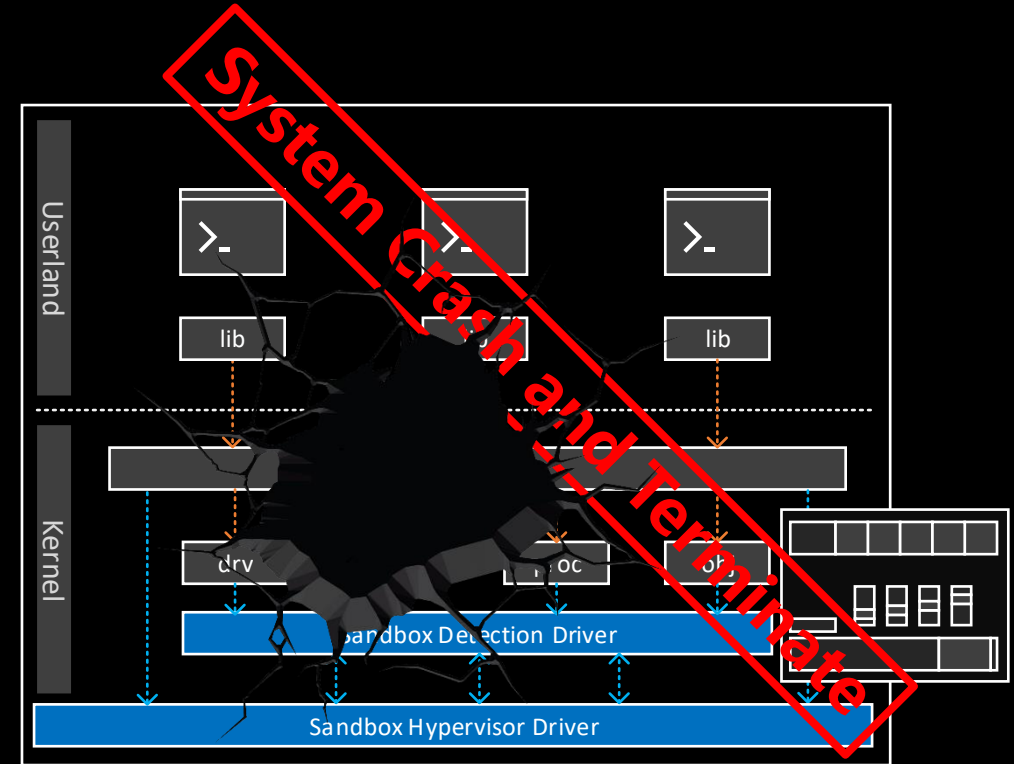
# Sandbox Detection Engine

- The 3<sup>rd</sup> Option: Virtualization-based Driver
- Virtualization-based system call monitoring
- R/W access to sensitive memory monitoring
- Avoid BSOD caused by PATCH GUARD 😊
- Protect private driver code and data 😊
- Expand more comprehensive detection 😊
- **Is this foolproof?**



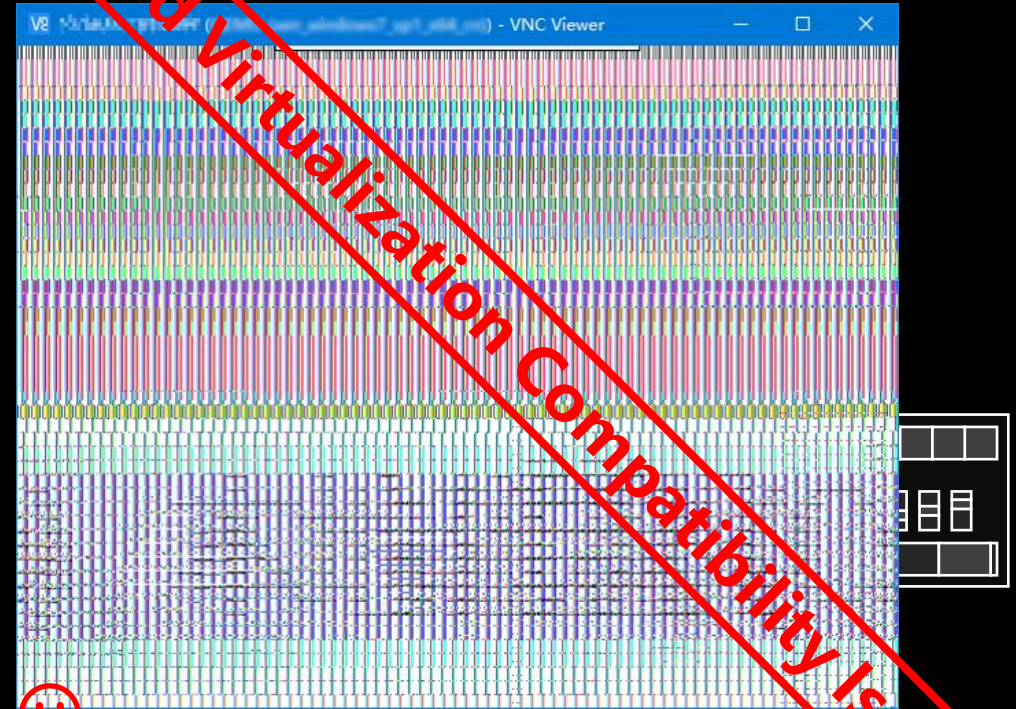
# Sandbox Detection Engine

- The 3<sup>rd</sup> Option: Virtualization-based Driver
- Virtualization-based system call monitoring
- R/W access to sensitive memory monitoring
- Avoid BSOD caused by PATCH GUARD 😊
- Protect private driver code and data 😊
- Expand more comprehensive detection 😊
- **Is this foolproof?**
- Unsecured reliability of other kernel modules 😞



# Sandbox Detection Engine

- The 3<sup>rd</sup> Option: Virtualization-based Driver
- Virtualization-based system call monitoring
- R/W access to sensitive memory monitoring
- Avoid BSOD caused by PATCH GUARD 😊
- Protect private driver code and data 😊
- Expand more comprehensive detection 😊
- **Is this foolproof?**
- Unsecured reliability of other kernel modules 😞
- Poor nested virtualization support of Virtual machine software 😞

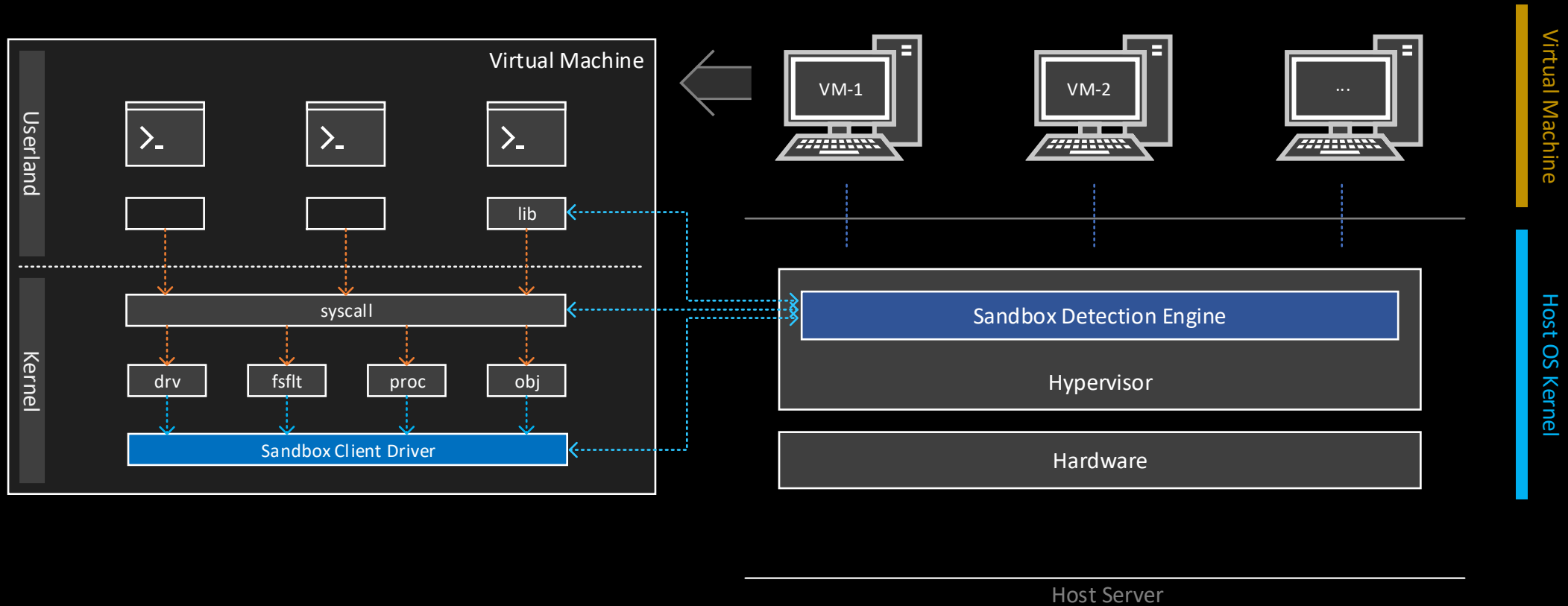


**Nested Virtualization Compatibility Issues**



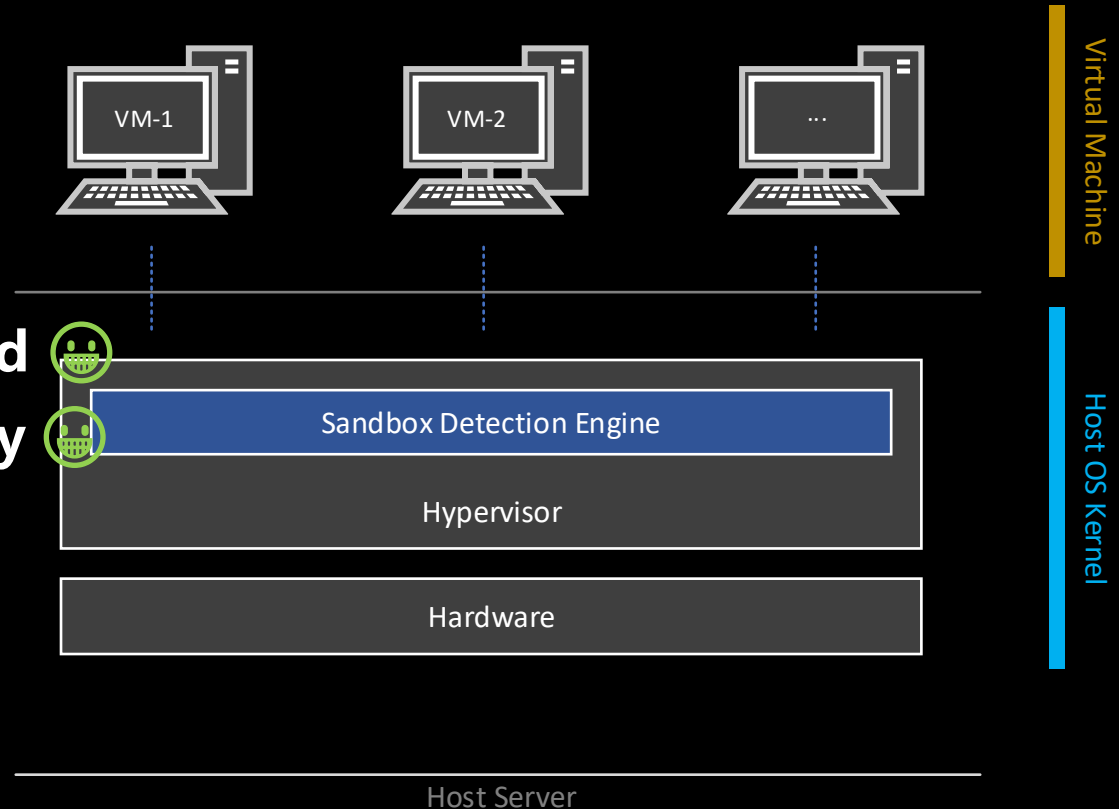
# Sandbox Detection Engine

- The 4<sup>th</sup> Option: Detection Scheme Based on Global Virtual Machine Monitor



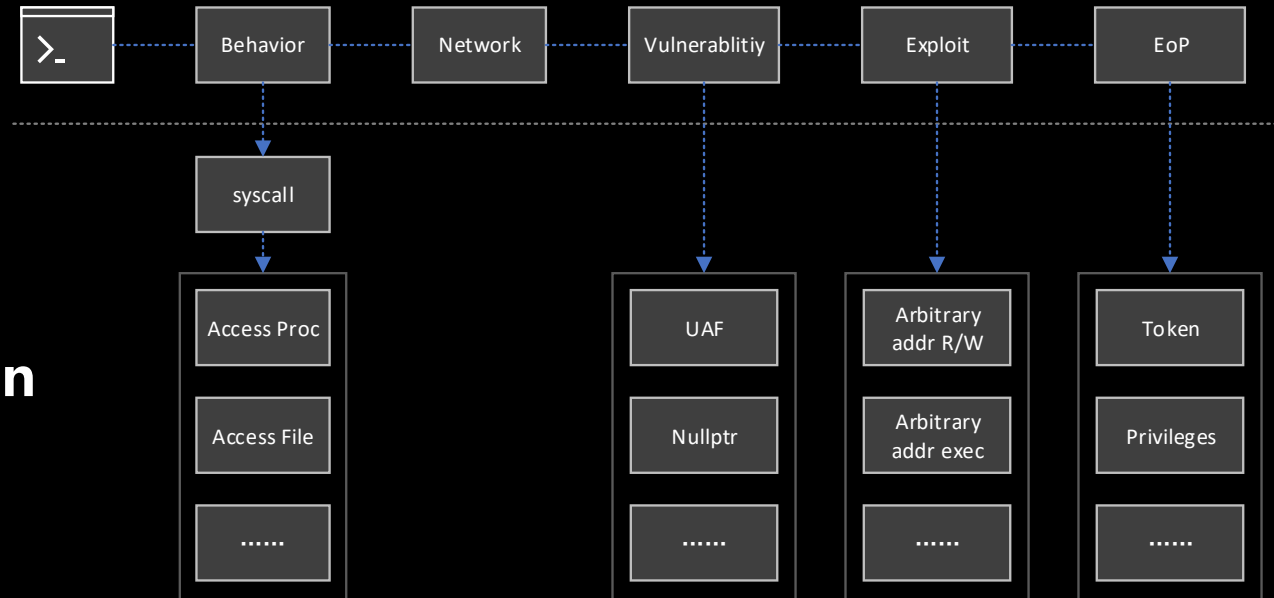
# Sandbox Detection Engine

- The 4<sup>th</sup> Option: Detection Scheme Based on Global Virtual Machine Monitor
- Core detection code in host OS kernel
- Integrated Advantages from previous
- Independent of modules inside VM 🤖
- No affect on detection when VM crashed 🤖
- Data outputs host record service directly 🤖



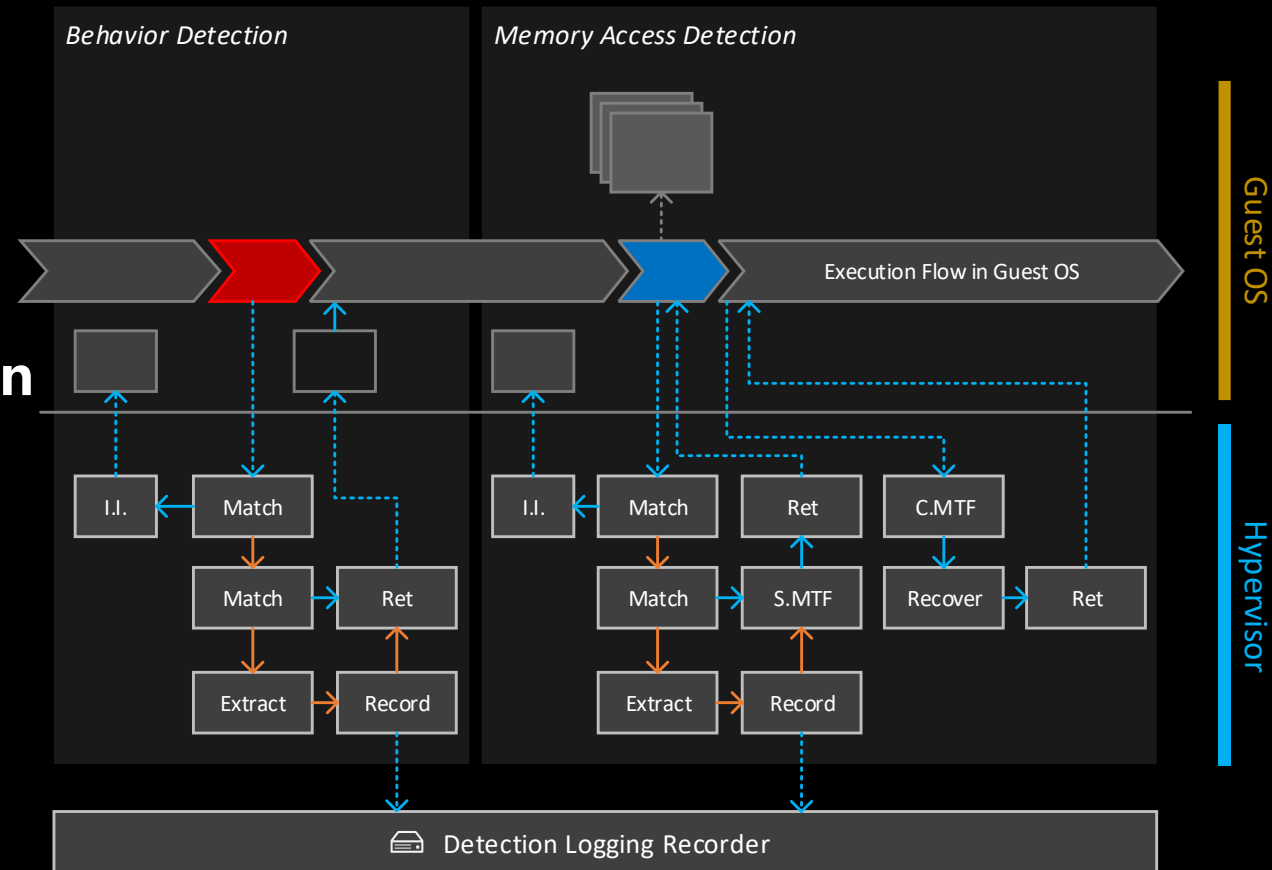
# Sandbox Detection Technology

- Behavior Detection
- Memory Access Detection
- Kernel Exploit Detection
- Kernel Exception Detection
- Known Vulnerabilities Detection
- User-mode Exploit Detection



# Sandbox Detection Technology

- Behavior Detection
- Memory Access Detection
- Kernel Exploit Detection
- Kernel Exception Detection
- Known Vulnerabilities Detection
- User-mode Exploit Detection



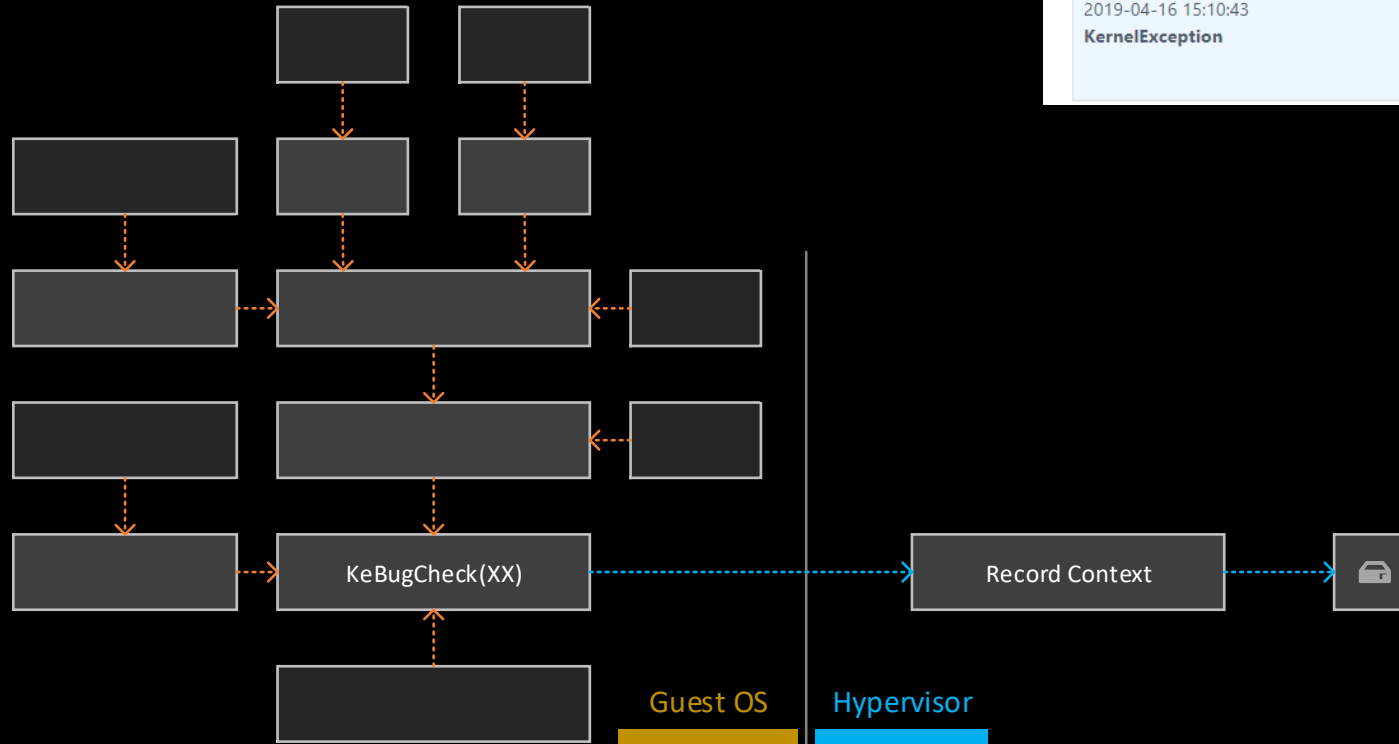
# Kernel Exploit Detection



Vulnerability Triggering	Exploiting	Exploit Result
UAF	Pool/heap spray	Token
Nullptr	Corrupting window	Privileges
OOB	...	Integrity
...		ACL
		...

# Kernel Exception Detection

- Record critical context when the system kernel crashes



Vulnerability and Exploit

Detected blue screen of death (BSOD) happened in the system (1 event)

Event	Context
2019-04-16 15:10:43 <b>KernelException</b>	bugcheck_code: 0xc2 parameter_1: 0x7 parameter_2: 0x1097 parameter_3: 0x5a080063 parameter_4: 0xfe793320

# Known Vulnerabilities Detection



- Identify tasks that exploit known vulnerabilities accurately

Vulnerability and Exploit

Detected process hit a kernel hotpatch (1 event)

Event	Context
2019-04-16 06:28:04 <b>KernelHotPatch</b>	checksum: 0x2486d5 module_base: 0x8f8b0000 cve_id: CVE-2018-0817 module_path: C:\WINDOWS\SYSTEM32\WIN32K.SYS

Vulnerability and Exploit

Detected process hit a user hotpatch (2 events)

Event	Context
2019-04-23 19:34:36 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000 cve_id: CVE-2018-0802 module_path: C:\Program Files\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE

# User-mode Exploit Detection



- **Heap Spray Limit Detection**
- **Export Address Table Filtering**
- **Import Address Table Filtering**
- **ROP Detection**
- **Flash Specific Detection**
  - Vector Length Detection
  - ByteArray Length Detection
  - LoadBytes Dump
  - Other Detection Features
- **VBScript Specific Detection**
- .....



# Detection Result Alarm



Computer DLLs are loaded or loaded (1 event)

### Information Stealing

- Contacting Microsoft Update, or the system (Microsoft Lock, DiskEncryption, SystemInformation) (1 event)
- Checking if Windows updates are installed (updates) (1 event)
- Checking if copy-paste features are installed (1 event)
- Requesting for user's public equipment key (1 event)
- Performing information about installed applications programs (1 event)

### Microsoft Software

- Modifying registry values (1 event)
- Performing Microsoft Updates (1 event)

### Networking

- Obtaining IP key value (1 event)

### Persistence / Autorun

- Performing registry values (1 event)


### Vulnerability and Exploit

- Detected process hit a user hotpatch (1 event)

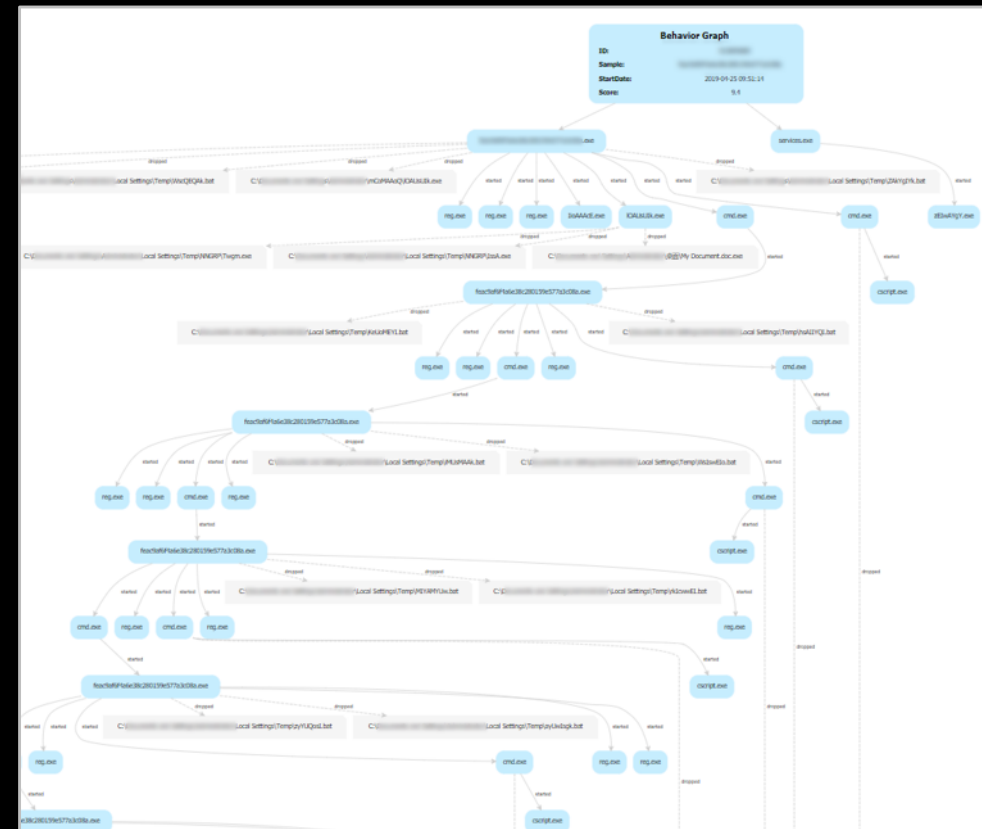
Event	Context
2019-04-25 15:34:45 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000 module_name: C:\Program Files\Microsoft Shared\EQUATION\EQNET32.EXE cve_id: CVE-2018-0802 module_path: C:\Program Files\Microsoft Shared\EQUATION\EQNET32.EXE

### Generic Behavior

- Modifying registry values (1 event)
- Performing registry values (1 event)
- Performing registry values (1 event)



9.4  
Malicious



# Detection Result Alarm



每日告警

2019-04-26

MDS	Origin Name	First Seen	Task ID	OS Environment	Qex Type	Tag	Score
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]

Advanced Threat Automation Platform

**How to find zero-day using sandbox?**

**Speaking from CVE-2017-0199...**

# Sandbox Advantage



- **Multiple Environments**
  - Each version of Windows
  - Each version of Office
  - Each version of Flash
- **Dynamic Execution**
  - Analog interaction
  - Anti-static obfuscation (especially RTF files)
- **Record And Restore The Scene**
- **Accurate**
  - Vulnerability and exploit identification
- **Automation**
  - Automatically show process behaviors
  - Automatically dump files
  - Automatically dump exploit code loaded by LoadBytes

# Build Automation Detection System



- **Historical Event Research**
  - History 0day/1day study
- **Data Source**
  - Massive data from 360
  - High quality shared data source
- **Analysis System**
  - Sandbox
- **Notification System**
- **Manual Confirmation**
  - Related Vulnerability Analysts

# Related Vulnerabilities in Nearly 6 Years



2013	2014	2015	2016	2017	2018
CVE-2013-0634 CVE-2013-3906	CVE-2014-1761 CVE-2014-4114 CVE-2014-6352	CVE-2015-1642 CVE-2015-2424 CVE-2015-2545 CVE-2015-5119 CVE-2015-5122	CVE-2016-4117 CVE-2016-7193 CVE-2016-7855	CVE-2017-0199 CVE-2017-0261 CVE-2017-0262 CVE-2017-8570 CVE-2017-8759 CVE-2017-11292 CVE-2017-11826 CVE-2017-11882	CVE-2018-0798 CVE-2018-0802 CVE-2018-4878 CVE-2018-5002 CVE-2018-8174 CVE-2018-8373 CVE-2018-15982

# Historical Vulnerability Classification



RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	ActiveX Control Parsing Problem	Office Embedded Flash 0day
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2012-1856 CVE-2015-2424 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 Code from HackingTeam CVE-2016-4117 CVE-2016-7855 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982
TIFF Image Parsing Problem	EPS File Parsing Problem	Moniker	Other Office Logic Vulnerabilities
CVE-2013-3906	CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759 CVE-2018-8174 CVE-2018-8373	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097



# History is Always Similar



RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	ActiveX Control Parsing Problem	Office Embedded Flash 0day
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 <b>CVE-2017-11826</b>	CVE-2012-0158 CVE-2012-1856 CVE-2015-2424 CVE-2017-11882 CVE-2018-0798 <b>CVE-2018-0802</b>	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 Code from HackingTeam CVE-2016-4117 CVE-2016-7855 CVE-2018-4878 <b>CVE-2018-5002</b> <b>CVE-2018-15982</b>
TIFF Image Parsing Problem	EPS File Parsing Problem	Moniker	Other Office Logic Vulnerabilities
CVE-2013-3906	CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759 <b>CVE-2018-8174</b> CVE-2018-8373	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

# Constant Reflection

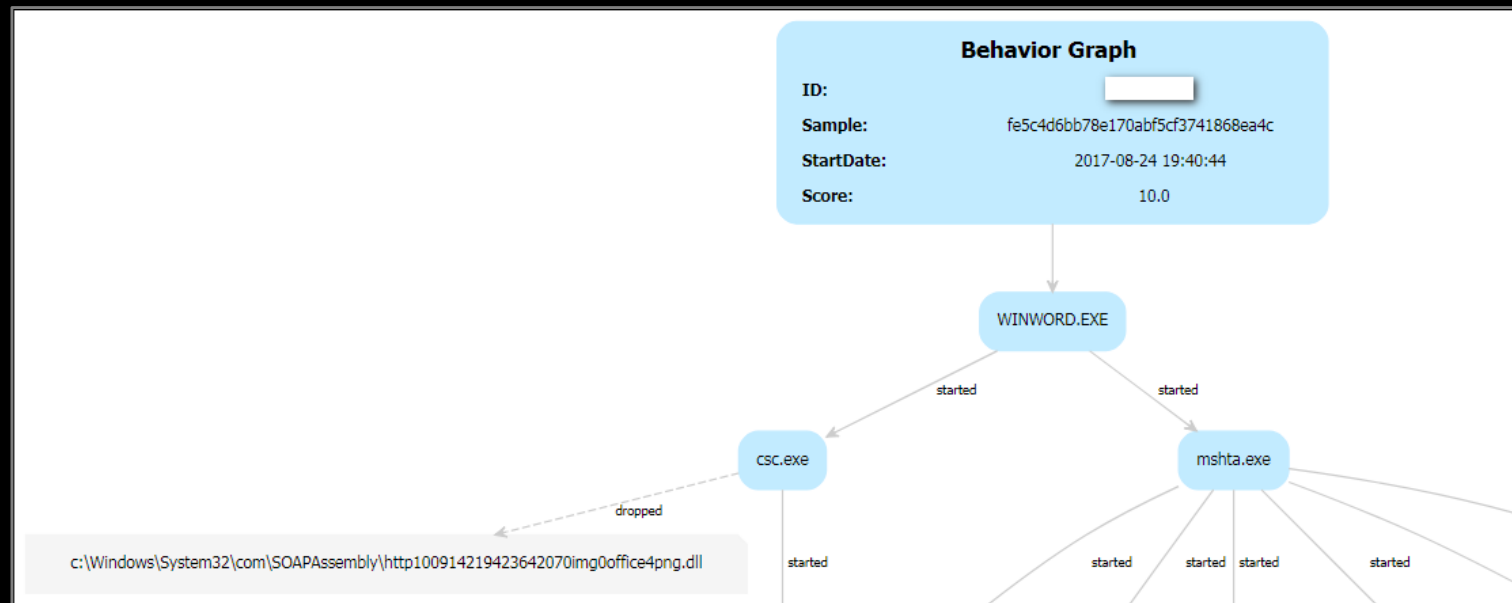
**A few missteps: 4 0days + 1 1day**

- **CVE-2017-0261 (0day)**
- **CVE-2017-0262 (0day) + CVE-2017-0263 (0day)**
  
- **Reflection**
  - Sandbox Detection Engine is defective 😞
  - CVE-2017-0261 sample cannot be triggered in Office 2010 😞
  - CVE-2017-0262 sample cannot be triggered in Office 2007 😞
  - When the user-mode engine meets a kernel zero-day 😞

- **CVE-2017-8759 (0day)**

- **Reflection**

- The sandbox ran out of the sample, but failed to notify the analyst in time 😞



- **CVE-2017-11292 (1day)**

- **Reflection**

- Lack of understanding of the DealersChoice framework 😞
- If the target is a low version of Flash, issue CVE-2015-7645 😞
- If the target is a high version of Flash, issue CVE-2017-11292 🧐

# Research Attack Framework



- **DealersChoice**
  - Named by @Unit42\_Intel
  - Used by APT28
  - Continuous improvement to avoid detection as much as possible
- **Initial Approach**
  - Check current Flash version
  - Filter geographical location
  - Short survival time
- **New Approach**
  - Anti-sandbox: need to simulate document slide
  - Rewrite open source code, add malicious features, avoid static detection

# Continue to Innovate

- **Sandbox Detection Engine defects** 😞
  - Develop the next generation of sandbox detection engine 😊
- **Correctly environment is not selected** 😞
  - Make a variety of environments 😊
  - Make delivery strategies with high trigger rate 😊
- **Failure to notify analysts in a timely manner** 😞
  - Build a real-time notification system 😊
- **Not familiar enough with the attack framework** 😞
  - Research DealersChoice framework 😊
  - Enhance Flash specific detection 😊

**From 0 to 1**

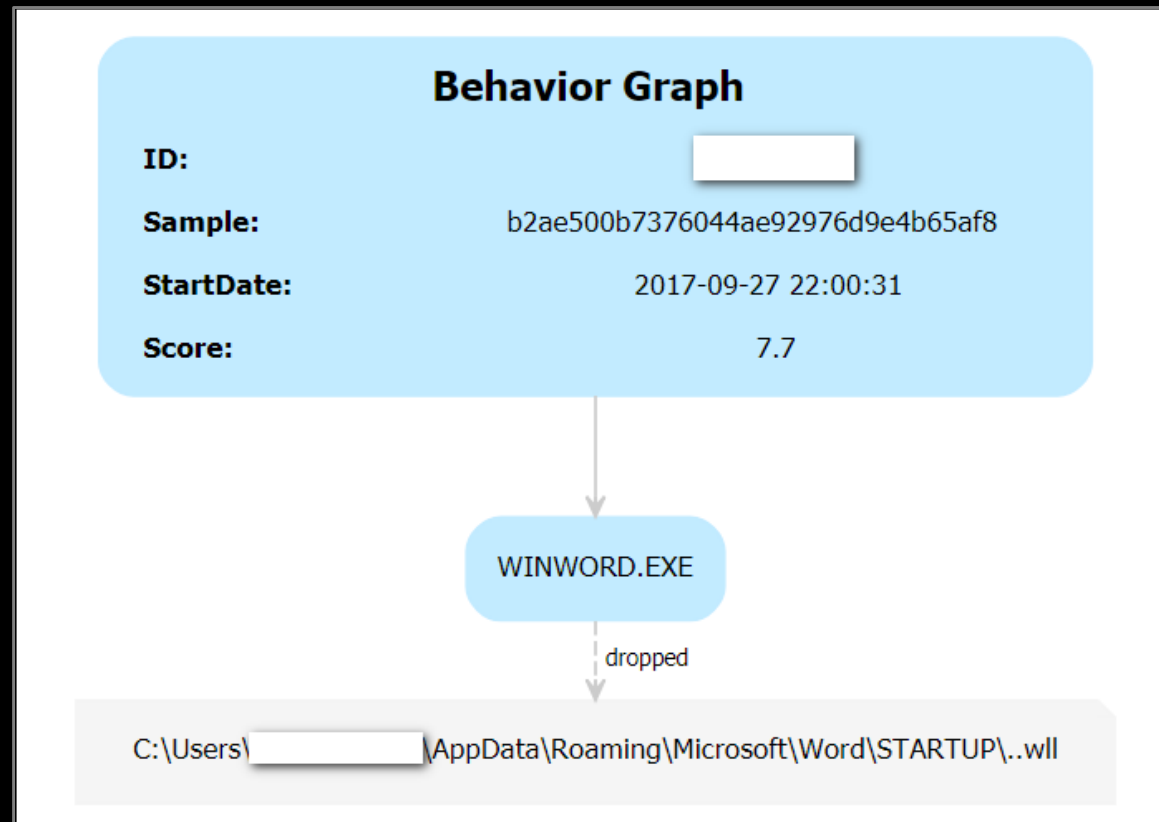


**CVE-2017-11826**



# From 0 to 1

- September 27, 2017



# From 0 to 1



- For the first time Chinese security company caught an in-the-wild Office zero-day

**Haifei Li** @HaifeiLi · 10 Oct 2017

Wow Qihoo360 detects an Office 0day attack ITW, a fair the 1st time for Chinese security companies? [twitter.com/360CoreSec/sta...](https://twitter.com/360CoreSec/status/918111111)

**360 Core Security** @360CoreSec  
Qihoo 360 Core Security detected an in-the-wild attack that leveraged CVE-2017-11826, an office 0day vulnerability..[goo.gl/VDmhF5](https://goo.gl/VDmhF5)

3 replies 24 retweets 33 likes

- OLEObject & Font object type obfuscation + ActiveX heap spray

```
; Normal execution under Office 2007

; mov     eax, [eax+44h]
0:000> dc 38450f4 l4c/4
038450f4 0000ffff 0000ffff 00000004 00000004 .....
03845104 00000001 00000000 00000000 00000000 .....
03845114 00000000 ffffffff ffffffff 00000000 .....
03845124 00000000 ffffffff 00000000 00000000 .....
03845134 00000000 01d9ffa0 67a02e58 .....X..g

; mov     eax, [eax+44h]
0:000> dc 01d9ffa0 l4c/4
01d9ffa0 00000001 00000001 01f47928 00000009 .....(y.....
01d9ffb0 00000000 00000000 00000000 00000000 .....
01d9ffc0 00000000 000004b0 00000000 00000000 .....
01d9ffd0 0005003c 00000000 00000000 00000000 <.....
01d9ffe0 00000002 01f7e0a0 00000000 .....

; mov     ecx, [eax]
0:000> dd 01f7e0a0 l1
01f7e0a0 65d9420c

; call    dword ptr [ecx+4]
0:000> dds 65d9420c l2
65d9420c 65b527ad mso!Ordinal1072+0x2dd
65d94210 658bbe71 mso!Ordinal1836+0xaf // AddRef
```

```
; Vuln triggered under Office 2007

; mov     eax, [eax+44h]
0:000> dc 5998140 l4c/4
05998140 000001de 000000dd 00000015 00000010 .....
05998150 00000000 00000000 00000000 00000000 .....
05998160 00000000 ffffffff ffffffff 00000000 .....
05998170 00000000 ffffffff 00000000 00000000 .....
05998180 00000000 04131700 67110a89 .....g

; mov     eax, [eax+44h]
0:000> dc 04131700 l4c/4
04131700 0000045f 00000000 00000000 00000000 _.....
04131710 00000000 00000000 00000000 00000000 .....
04131720 00000000 00000000 0069004c 0063006e .....L.i.n.c.
04131730 00720065 00680043 00720061 00680043 e.r.C.h.a.r.C.h.
04131740 00720061 088888ec 006f0066 a.r.....f.o.

; mov     ecx, [eax]
0:000> dd 088888ec l1
088888ec 088888ec

; call    dword ptr [ecx+4]
0:000> dds 088883ec l2
088883ec 72980e2b MSVBVM60!IID_IVbaHost+0x127eb
088883f0 72980e2b MSVBVM60!IID_IVbaHost+0x127eb // Stack Pivot
```

# From 1 to N



**CVE-2018-0802**

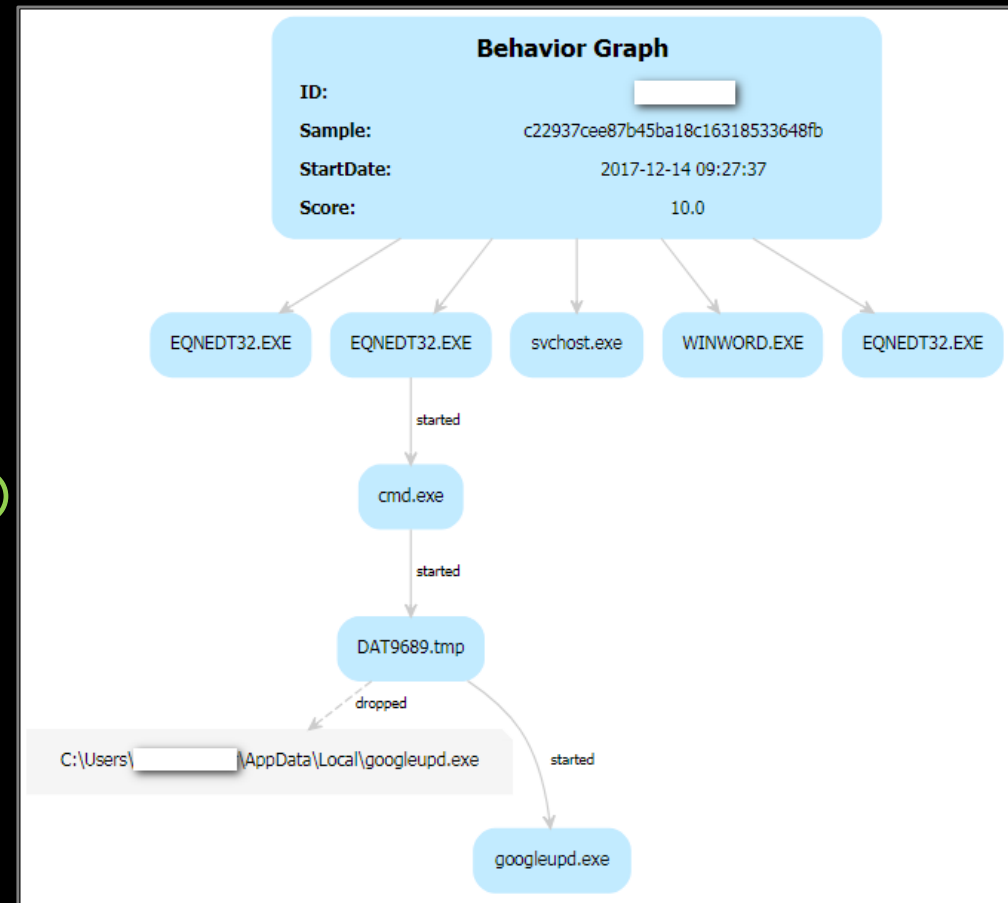
**CVE-2018-8174**

**CVE-2018-5002**

**CVE-2018-15982**

# CVE-2018-0802

- **Stack Overflow in Equation Editor**
- **December 14, 2017**
- **Embedding two vulnerabilities**
  - CVE-2017-11882
  - CVE-2018-0802
  - Can be triggered and exploited successfully 😊
- **December 19, 2017**
- **Embedding only one vulnerability**
  - CVE-2018-0802
  - Cannot trigger properly 😞
  - Can be successfully used after reconstructing OLE 😊



# CVE-2018-0802



- Both samples were reported to Microsoft
- On January 10, 2018, Microsoft acknowledged us

## Acknowledgements

Luka Treiber of Opatch Team - ACROS Security

Netanel Ben Simon and Omer Gull of Check Point Software Technologies

Liang Yin of Tencent PC Manager

zhouat of Qihoo 360 Vulcan Team

Zhiyuan Zheng

Yuki Chen of Qihoo 360 Vulcan Team

Yang Kang, Ding Maoyin and Song Shenlei, and Jinquan of Qihoo 360 Core Security (@360CoreSec)

bee13oy of Qihoo 360 Vulcan Team



# CVE-2018-0802



- **Where is the error?**
  - Extract the confusing OLE object

```
0:010> bp ole32!OleConvertOLESTREAMToIStorage
0:010> g
Breakpoint 0 hit
eax=000004e0 ebx=059bc3c0 ecx=00008000 edx=00000000 esi=02d80960 edi=001dade8
eip=75c528fa esp=001dab2c ebp=001dadab iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00200206
ole32!OleConvertOLESTREAMToIStorage:
75c528fa 8bff          mov     edi,edi
0:000> .writemem C:\de-obfuscated_ole.bin poi(poi(poi(esp + 0x04) + 0x08)) Lpoi(poi(esp + 0x04) + 0x0C)
Writing dc5 bytes..


0:000> db poi(poi(poi(esp + 0x04) + 0x08))
04946510  01 05 00 00 02 00 00 00 00-0b 00 00 00 45 71 75 61  .....Equa
04946520  74 69 6f 6e 2e 33 00 00-00 00 00 00 00 00 00 00  .....tion.3.....
04946530  0e 00 00 d0 cf 11 e0 a1-b1 1a e1 00 00 00 00 00 00  .....
04946540  00 00 00 00 00 00 00 00-00 00 00 00 3e 00 03 00 fe  .....>....
04946550  ff 09 00 06 00 00 00 00-00 00 00 00 00 00 00 01  .....
04946560  00 00 00 01 00 00 00 00-00 00 00 00 10 00 00 02  .....
04946570  00 00 00 01 00 00 00 fe-ff ff ff 00 00 00 00 00 00  .....
04946580  00 00 00 ff ff ff ff ff-ff ff ff ff ff ff ff  .....

```



# CVE-2018-0802

- Where is the error?
  - MiniFat Sector misaligned 0x15 bytes



```
05E0h: 00 00 00 06 00 00 00 00 00 00 00 FE FF FF FF 02 00 00 00 FE FF FF FF FE FF FF FF 05 00 00 00 06
0600h: 00 00 00 07 00 00 00 FE FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0620h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0640h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0660h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0680h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
06A0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
06C0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
06E0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0700h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0720h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0740h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0760h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0780h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
07A0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
07C0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
07E0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0800h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0820h: 00 00 00 00 00 00 00 00 00 00 00 01 00 FE FF 03 0A 00 00 FF FF FF FF 02 CE 02 00 00 00 00 00 C0
```

Template Results - OLESS.bt


Name	Value
> struct OLESSHEAD OleHeader	
> SECT Fat[128]	
> SECT MiniFat[128]	



# CVE-2018-0802



- After the New Year's Day in 2018, more CVE-2018-0802 samples appeared
- Other researchers noticed the samples but didn't know they used a zero-day 😞


 **@blu3\_team** @blu3\_team · 6 Jan 2018

#Malware using Word add-in persistence  
Sample uses the CVE-2017-11882 %temp% dropper method to  
%APPDATA%\Microsoft\word\startup\w.wll

@MalwareParty #infosec

[mymalwareparty.blogspot.com/2018/01/word-a...](http://mymalwareparty.blogspot.com/2018/01/word-a...)


1 33 51

 **jq0904** @jq0904

@blu3\_team Have a look at this article, you may miss a 0day few days ago

**b0ring** @dnpushme  
Want to know how cve-2018-0802 bypass alsr 😊?  
Read this analysis [anquanke.com/post/id/94210](http://anquanke.com/post/id/94210)

12:14 AM - 10 Jan 2018

1 Like 

# How to Distinguish Two Vulnerabilities



```
IPersistStorage::Load(406881)
  offset:406a93    call ReadMTEFData(42f8ff)
    offset:42f921    call 43755c
      offset:4375d5    call 43a720
        offset:43a72a    call 43a87a
          offset:43a89b    call 43b418
            ; Font tag parse Logic
            offset:43b44b    call ReadFontName(4164fa)
            offset:43b461    call 4214c6
              offset:4214dd    call LogfontStruct_Overflow(421774)
                offset:4217c3    call 421e39
                  offset:421e5e    rep movsd <- CVE-2018-0802
                offset:4218cb    call 451d50
                offset:4218df    call 4115a7
                  offset:4115d3    call final_overflow(4115d3)
                    offset:411658    rep movsd <- CVE-2017-11882
                    offset:411874    retn
```

# How to Distinguish Multiple Vulnerabilities



- Accurately distinguish between the three equation editor vulnerabilities

2018-01-24 01:28:57 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000  cve_id: CVE-2017-11882 module_path: C:\Program Files\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE
2018-09-27 12:37:34 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000  cve_id: CVE-2018-0798 module_path: C:\Program Files\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE
2018-03-30 21:30:29 <b>UserHotPatch</b>	checksum: 0x874f7 module_base: 0x8e0000 cve_id: CVE-2018-0802  module_path: C:\Program Files\*\Microsoft Shared\EQUATION\EQNEDT32.EXE

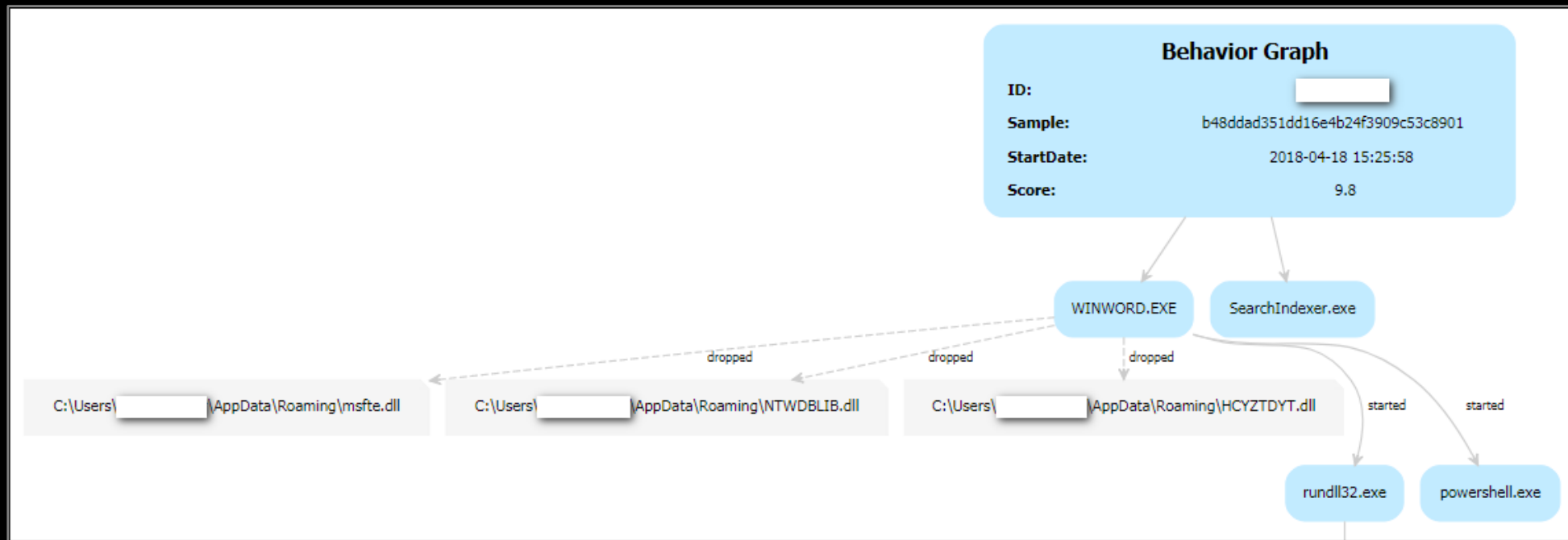
# CVE-2018-8174



- **Two earlier Office samples**
- **Moniker remote loading CVE-2014-6332**
  
- **January 17, 2018**
  - Document MD5: A9D3F7A1ACD624DE705CF27EC699B6B6
  - Moniker: `hxxp://s.dropcanvas[.]com/1000000/940000/939574/akw.html`
  - `akw.html` MD5: C40A128AE7AEFFA3C1720A516A99BBDF
  
- **February 23, 2018**
  - Document MD5: 2E658D4A286F3A4176A60B2450E9E729
  - Moniker: `hxxp://s.dropcanvas[.]com/1000000/942000/941030/IE.html`
  - `IE.html` MD5: C36D544588BAF97838588E732B3D47E9

# CVE-2018-8174

- April 18, 2018
- RTF document loading and executing VBScript zero-day



# CVE-2018-8174



- On May 8, 2018, Microsoft acknowledged us

## Acknowledgements

Anonymous working with Trend Micro's Zero Day Initiative

Vladislav Stolyarov of Kaspersky Lab

Yang Kang of Qihoo 360 Core Security

Ding Maoyin of Qihoo 360 Core Security

Dan Lutas of Bitdefender

Anton Ivanov of Kaspersky Lab

Simon Zuckerbraun working with Trend Micro's Zero Day Initiative

Jinquan of Qihoo 360 Core Security

Song Shenlei of Qihoo 360 Core Security



# CVE-2018-8174



- UAF -> overlength array -> arbitrary address read and write

```
Class class_setprop_a
  Dim mem

  Function P
  End Function

  Function SetProp(Value)
    mem = Value 'callback
    SetProp = 0
  End Function
End Class
```

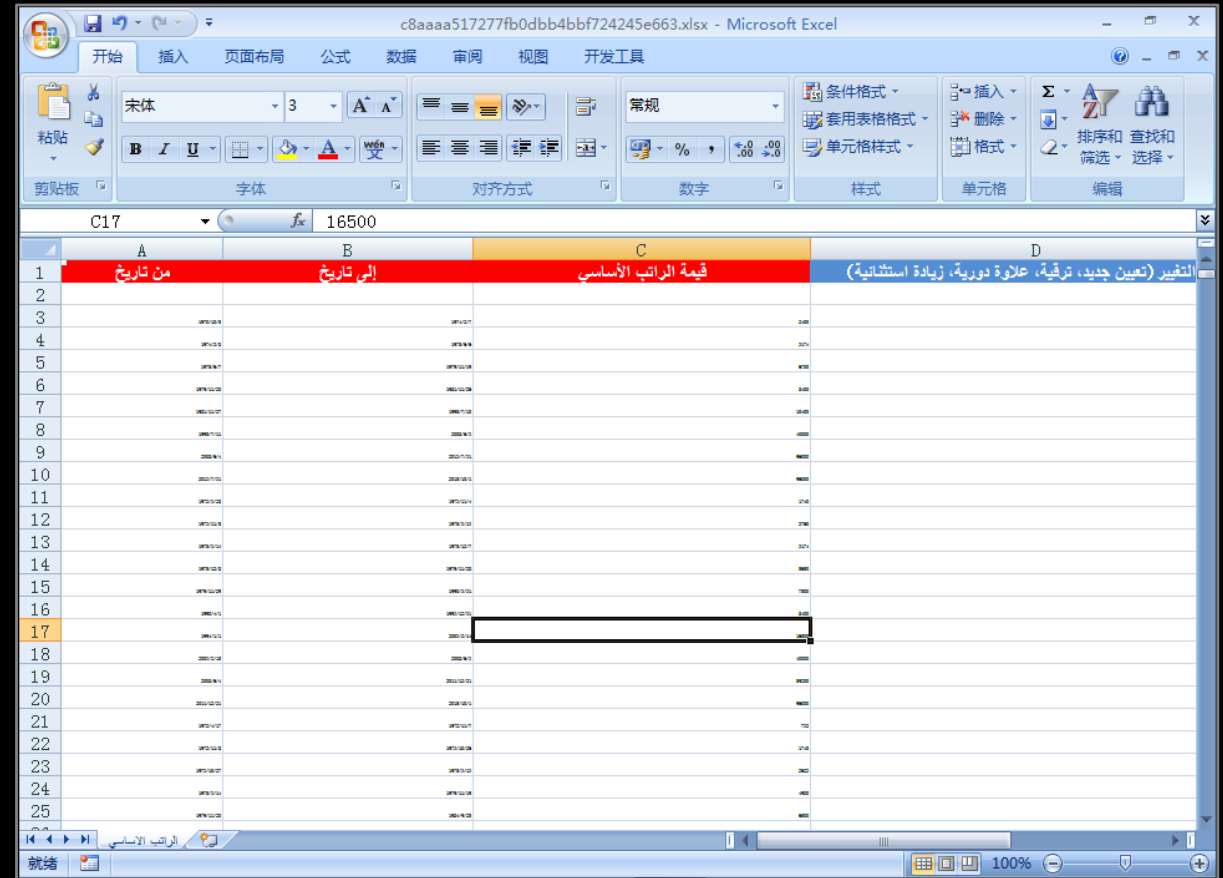
```
// before mem = Value
0:005> dd 022cb91c 14
022cb91c  00000008 00000000 04730834 00000000
0:005> dd 04730834 16
04730834  08800001 00000001 00000000 00000000
04730844  7fffffff 00000000

// after mem = Value
0:007> dd 022cb91c 14
022cb91c  0000200c 00000000 04730834 00000000
0:007> dd 04730834 16
04730834  08800001 00000001 00000000 00000000
04730844  7fffffff 00000000
```

# CVE-2018-5002



- June 1, 2018
- A complex Flash control framework
- AVM2 Interpreter Vulnerability



# CVE-2018-5002



- On June 7, 2018, Adobe acknowledged us

• CVE-2018-5002 was independently identified and reported by the following organizations and individuals: Chenming Xu and Jason Jones of ICEBRG, Bai Haowen, Zeng Haitao and Huang Chaowen of 360 Threat Intelligence Center of 360 Enterprise Security Group, and Yang Kang, Hu Jiang, Zhang Qing, and Jin Quan of Qihoo 360 Core Security (@360CoreSec), Tencent PC Manager (<http://guanjia.qq.com/>)

# CVE-2018-5002

- Bypass ROP detection 😊
- Override return address to bypass CFG 😊
- Unable to bypass EAF detection 😞

```
var cls25:class_25 = new class_25(cls8, RtlUnwind_Addr);
var NtProtectVirtualMemory_Addr:uint = cls25.GetFuncAddrByEAT("NtProtectVirtualMemory");
if(0 == NtProtectVirtualMemory_Addr)
{
    return new Array();
}

var NtPrivilegedServiceAuditAlarm_Addr:uint = cls25.GetFuncAddrByEAT("NtPrivilegedServiceAuditAlarm");
if(0 == NtPrivilegedServiceAuditAlarm_Addr)
{
    return new Array();
}
```

# How to Debug CVE-2018-5002

- Reverse -> ASC2.0 Compile -> Modify bytecode with FFDEC -> Debuggable swf file
- Open source WinDBG plugin
  - [https://github.com/michaelpdu/flashext\\_pykd](https://github.com/michaelpdu/flashext_pykd)
- Add 3 lines of code to make the plugin more perfect 🤖

```
def callback_after_call_getmethodname(self):
    # dprintln("Enter into callback_after_call_getmethodname")
    reg_eax = reg("eax")
    # dprintln("EAX = " + hex(reg_eax))
    addr_name = ptrPtr(reg_eax + 0x08)
    len_name = ptrPtr(reg_eax + 0x10)

    if 0 == addr_name and 0 != len_name:
        if ptrPtr(reg_eax + 0x0C) != 0:
            addr_name = ptrPtr(ptrPtr(reg_eax + 0x0C) + 0x08)
```

# CVE-2018-5002 in the Debugger



- Trigger Vulnerability -> Swap Pointer on stack -> Type Confusion

```
// Before triggering
0:007> dd 02c0ab24-10
02c0ab14  093101f0 093101a0 093101f0 093101a0
02c0ab24  093101f0 093101a0 093101f0 093101a0
02c0ab34  093101f0 093101a0 093101f0 093101a0
02c0ab44  093101f0 093101a0 093101f0 093101a0
02c0ab54  093101f0 093101a0 093101f0 093101a0
02c0ab64  093101f0 093101a0 093101f0 093101a0
02c0ab74  093101f0 093101a0 093101f0 093101a0
02c0ab84  093101f0 093101a0 093101f0 093101a0

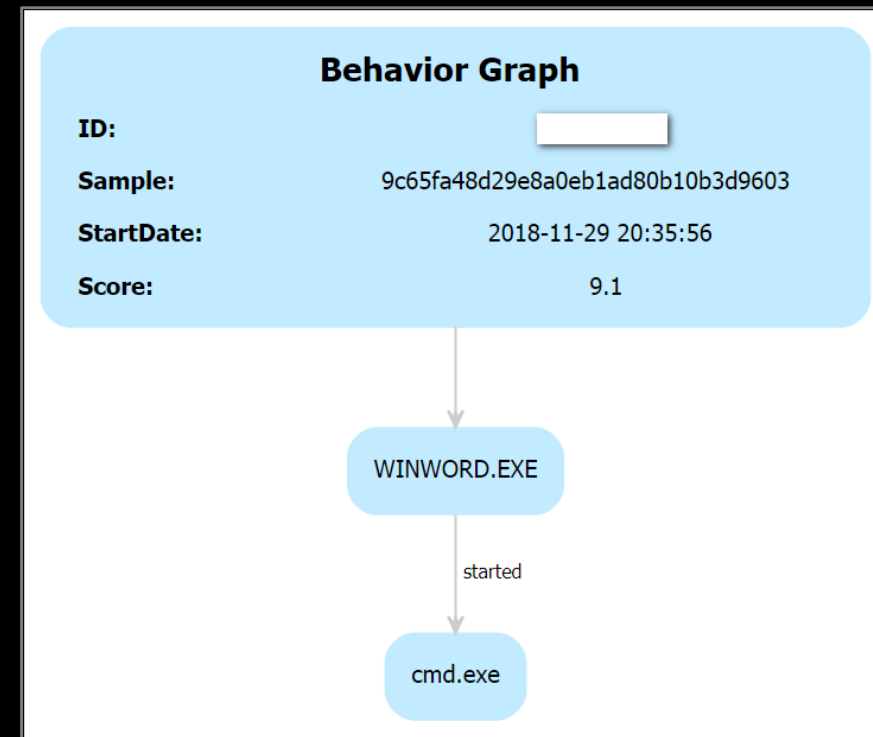
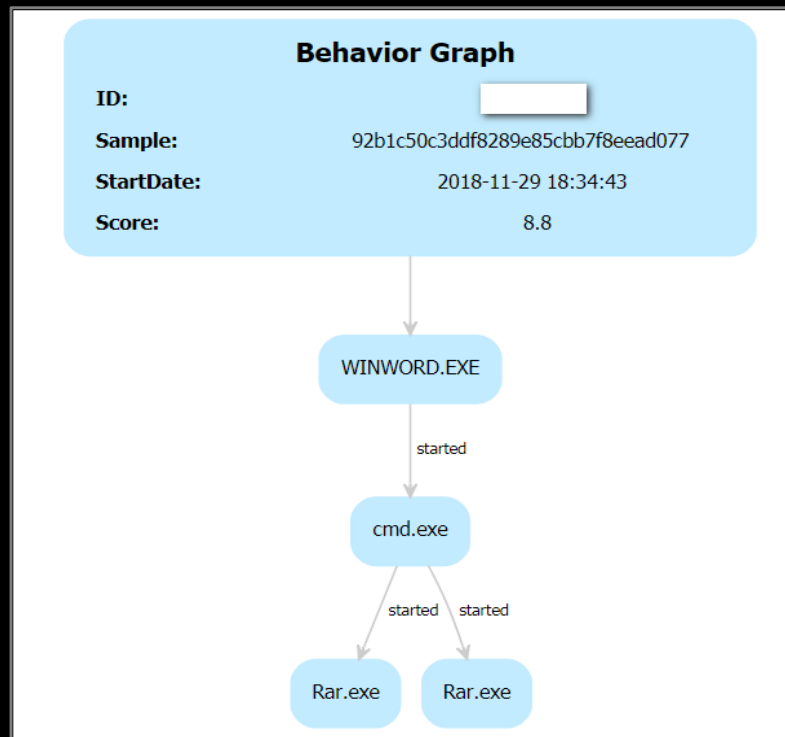
// After triggering
0:007> dd 02c0ab24-10
02c0ab14  093101f0 093101a0 093101f0 093101f0
02c0ab24  093101a0 093101a0 093101f0 093101a0
02c0ab34  093101f0 093101a0 093101f0 093101a0
02c0ab44  093101f0 093101a0 093101f0 093101a0
02c0ab54  093101f0 093101a0 093101f0 093101a0
02c0ab64  093101f0 093101a0 093101f0 093101a0
02c0ab74  093101f0 093101a0 093101f0 093101a0
02c0ab84  093101f0 093101a0 093101f0 093101a0
```

```
class_6
1 package
2 {
3     import avm2.intrinsics.memory.li8;
4
5     public class class_6
6     {
7
8         {
9             li8(123456);
10        }
11
12        public function class_6()
13        {
14            super();
15        }
16    }
17 }
18

1 method
2 name null
3 returns null
4
5 body
6 maxstack 3
7 localcount 2
8 initscopedepth 3
9 maxscopedepth 6
10 try from ofs0000 to ofs0004 target ofs0004
11
12 code
13 try{
14     jump ofs0024
15 }
16
17 catch{
18 ofs0004:
19     local_0 = local_449
20     local_449 = local_448
21     local_448 = local_0
22     jump ofs0028
23 }
24
25 ofs0024:
26     li8(123456)
27
28 ofs0028:
29     returnvoid
```

# CVE-2018-15982

- November 29, 2018
- 2 hours, 2 samples
- UAF Vulnerability in TVSDK



- On December 5, 2018, Adobe acknowledged us again

## Acknowledgments

Adobe would like to thank the following individuals and organizations for reporting the relevant issues and for working with Adobe to help protect our customers:

- Chenming Xu and Ed Miles of Gigamon ATR (CVE-2018-15982)
- [Yang Kang \(@dnpushme\) and Jinquan \(@jq0904\) of Qihoo 360 Core Security \(@360CoreSec\) \(CVE-2018-15982\)](#)
- He Zhiqiu, Qu Yifan, Bai Haowen, Zeng Haitao and Gu Liang of 360 Threat Intelligence of 360 Enterprise Security Group (CVE-2018-15982)
- b2ahex (CVE-2018-15982)



# CVE-2018-15982

- Use HackingTeam's trick to bypass ROP detection 😊
- Unable to evade EAF detection 😞

```
// Virt(ualPro)tect = 74726956 74636574
var vp_addr:uint = this.getFuncAddrByEAT32(0x74726956, 0x74636574, 10, kernel32_addr);

...

this.writeDWORD32(sc_addr + 8 + 0x80 + 0x1c, vp_addr);
this.writeDWORD32(ptbl, sc_addr + 8 + 0x80);
this.writeDWORD32(p + 0x1c, sc_addr);
this.writeDWORD32(p + 0x20, vec_uint.length * 4);
var args:Array = new Array(0x41);
Payload.call.apply(null, args); // Call VirtualProtect to bypass DEP
```

# Other Harvest



- 1 Word CVE 🍷
- 1 PowerPoint CVE 🍷
- 4 Excel CVE 🍷
- 1 Win32k CVE 🍷

Microsoft Excel Remote Code Execution Vulnerability	CVE-2018-0920	Yangkang (@dnpushme) & Wanglu of Qihoo360 CoreSecurity @360CoreSec Vladislav Stolyarov of Kaspersky Lab
Microsoft PowerPoint Remote Code Execution Vulnerability	CVE-2018-8376	yangkang(@dnpushme) & Jinquan(@jq0904) & Wanglu of Qihoo360 CoreSecurity(@360CoreSec)
Microsoft Excel Remote Code Execution Vulnerability	CVE-2018-8379	Jinquan(@jq0904) of Qihoo360 CoreSecurity(@360CoreSec) Yangkang(@dnpushme) of Qihoo360 CoreSecurity(@360CoreSec)
Microsoft Word Remote Code Execution Vulnerability	CVE-2018-8539	Yangkang of 360CoreSec Jinquan of 360CoreSec
Microsoft Excel Information Disclosure Vulnerability	CVE-2018-8627	Yangkang(@dnpushme) & Jinquan(@jq0904) of Qihoo360 CoreSecurity(@360CoreSec)
Microsoft Excel Information Disclosure Vulnerability	CVE-2019-0669	Jinquan of 360CoreSec Yangkang of 360CoreSec
Windows GDI Elevation of Privilege Vulnerability	CVE-2018-0817	HongZhenhao Li Qi(@leeqwind) of Qihoo 360

# Summary

- **Easy from 1 to N, hard from 0 to 1**
- **Know your opponent, reflect upon yourself, beat your opponent**
- **Always on the road**



# Acknowledgement



- **Thanks to all the partners of 360 Advanced Threat Team**
- **Thanks to @programmeboy, @guhe120, @binjo, @Unit42\_Intel**
- **Special thanks to @HaifeiLi and his sharing about Office security**

# BLUEHAT

## SHANGHAI 2019

### Needle in A Haystack: Catch Multiple Zero-days Using Sandbox

Qi Li    Quan Jin

[liqi3-s@360.cn](mailto:liqi3-s@360.cn)    [jinquan@360.cn](mailto:jinquan@360.cn)

[@leeqwind](#)    [@jq0904](#)