处理器 A 级漏洞 Meltdown(熔毁)和 Spectre(幽灵)分析报告

安天安全研究与应急处理中心（Antiy CERT）

安天微电子与嵌入式安全研发中心

报告初稿完成时间：2018 年 1 月 4 日 23 时
首次发布时间：2018 年 1 月 4 日 23 时
本版更新时间：2018 年 1 月 5 日 08 时 50 分

1 概述

安天应急处理中心在 2018 年 1 月 4 日，针对 Google 公司的 Project Zero 等安全团队披露出的**英特尔等处理器芯片存在非常严重的安全漏洞**，发布了 **A 级漏洞风险通告**，并提醒**该漏洞演化为针对云和信息基础设施的 A 级网络安全灾难**。相关漏洞**利用了芯片硬件层面执行加速机制的实现缺陷实现侧信道攻击，可以间接通过 CPU 缓存读取系统内存数据**。漏洞 Meltdown（熔毁）因“融化”了硬件的安全边界而得名，漏洞 Spectre（幽灵）因其手段的隐蔽性而得名。

该漏洞是一个足以动摇全球云计算基础设施根基的漏洞，其意味着任何虚拟机的租户或者入侵了成功一个虚拟机的攻击者，都可以通过相关攻击机制去获取完整的物理机的 CPU 缓存数据，而这种攻击对现有虚拟化节点的防御机制是无法感知的。同时由于**该漏洞的机理，导致其存在各种操作系统平台的攻击方式，因此尽管这一漏洞本身只能读取数据，不能修改数据，但由于其获取的数据中有可能包括口令、证书和其他关键数据，包括能够完整 Dump 内存镜像，因此这个漏洞比一般性的虚拟机逃逸对云的危害更大**。尽管当前全球主要云服务商均在积极应对这一漏洞的影响，但鉴于这些云服务体系的庞大而复杂，以及大面积补丁本身所面临的复杂度和风险，漏洞利用 POC 已经发布并验证成功，因此**这次漏洞修补已经成为一场时间赛跑**。在这个过程中，**攻击者所获取到的数据，将会沉淀出对于关键数据和隐私泄露、登陆凭证被窃取导致连锁攻击等次生灾害**。

鉴于大量政企机构和行业进行了私有云的建设，而私有云的安全防御和补丁升级可能更弱。因此后续**需要深度注意利用该漏洞在私有云中进行的攻击**。

同时，**该漏洞对于攻击桌面节点同样有巨大的攻击力，其大大提升了以浏览器等为攻击入口的攻击成功率**。包括使传统的使用非超级用户来降低网络风险的安全策略失效。

当前已经公布的漏洞 POC 对 Intel 系列 CPU 有效，但鉴于相关执行加速机制是现代处理器的通用技术，因此所有处理器均需要分析相关风险。

安天在第一时间向管理部门提交威胁通报，并根据管理部门的要求正在进一步做深度的分析验证和应对工作。并在 1 月 4 日发出了公开漏洞通告。鉴于相关漏洞机理较为复杂，涉及到体系结构、操作系统，特别是 CPU 的核心运行机制，为使主管部门和用户深入了解漏洞细节，做好防护，安天组织内部公益翻译和技术团队对披露这两个漏洞的长篇关键论文文献《Meltdown》和《Spectre》进行了翻译，并在 1 月 4 日发布了《Meltdown》的初稿，《Spectre》也在紧张翻译工作中。

2 漏洞介绍

针对英特尔处理器涉及到两种攻击方法，分别为 Meltdown 和 Spectre， Spectre 涉及 CVE 编号 CVE-2017-5753 和 CVE-2017-5715，而 Meltdown 涉及 CVE 编号 CVE-2017-5754。

Meltdown 破坏了位于用户和操作系统之间的基本隔离，此攻击允许程序访问内存，因此其他程序以及操作系统的敏感信息会被窃取。这个漏洞“熔化”了由硬件来实现的安全边界。允许低权限用户级别的应用程序“越界”访问系统级的内存，从而造成数据泄露。

Spectre 则是破坏了不同应用程序之间的隔离。问题的根源在于推测执行（speculative execution），这是一种优化技术，处理器会推测在未来有用的数据并执行计算。这种技术的目的在于提前准备好计算结果，当这些数据被需要时可立即使用。在此过程中，英特尔没有很好地将低权限的应用程序与访问内核内存分开，这意味着攻击者可以使用恶意应用程序来获取应该被隔离的私有数据。

3 影响范围

本次安全事件影响到的范围很广，包括：

处理器芯片：英特尔为主、ARM、AMD，对其他处理器同样可能存在相关风险。

操作系统：Windows、Linux、macOS、Android

云服务提供商：亚马逊、微软、谷歌、腾讯云、阿里云等

各种私有云基础设施。

桌面用户可能遭遇到结合该机理组合攻击。

4 漏洞分析

4.1 CPU 缓存验证缺陷

分支预测[6]和乱序执行[7]，是一种 CPU 优化技术，CPU 会执行一些可能在将来会执行的任务。当分支指令发出之后，无相关优化技术的处理器，在未收到正确的反馈信息之前，是不会做任何处理；而具有优化技术能力的新型处理器，可以预测即将执行的指令，会预先处理一些指令所需的数据，例如将下一条指令所需要访问的内存提前加载到 CPU 缓存中，这就避免了执行具体指令时再去读内存，从而加快了 CPU 的执行速度，具体流程如下所示：

图 1 CPU 优化执行

指令 3 如果出现问题（如指令 3 是一个除 0 或者是一个非法的操作），会触发 CPU 的异常处理，具体情况如下

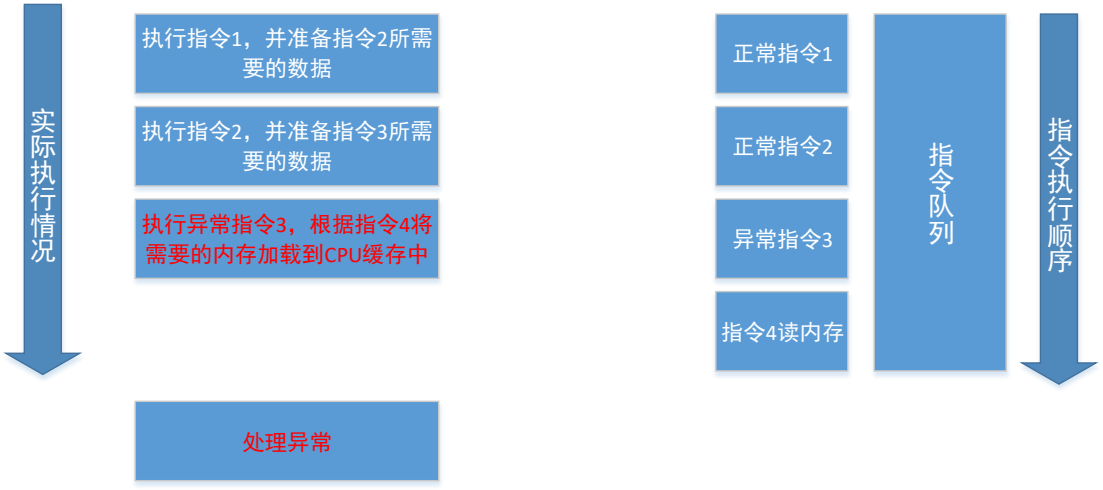

图 2 CPU 异常指令执行

对于具有预测执行能力的新型处理器，在实际 CPU 执行过程中，指令 4 所需的内存加载环节不依赖于指令 3 是否能够正常执行，而且从内存到缓存加载这个环节不会验证访问的内存是否合法有效。即使指令

3 出现异常，指令 4 无法执行，但指令 4 所需的内存数据已加载到 CPU 缓存中，这一结果导致指令 4 即使加载的是无权限访问的内存数据，该内存数据也会加载到 CPU 缓存中，因为 CPU 是在缓存到寄存器这个环节才去检测地址是否合法，而 CPU 分支预测仅仅是完成内存到 CPU 缓存的加载，实际指令 4 并没有被真正的执行，所以他的非法访问是不会触发异常的。

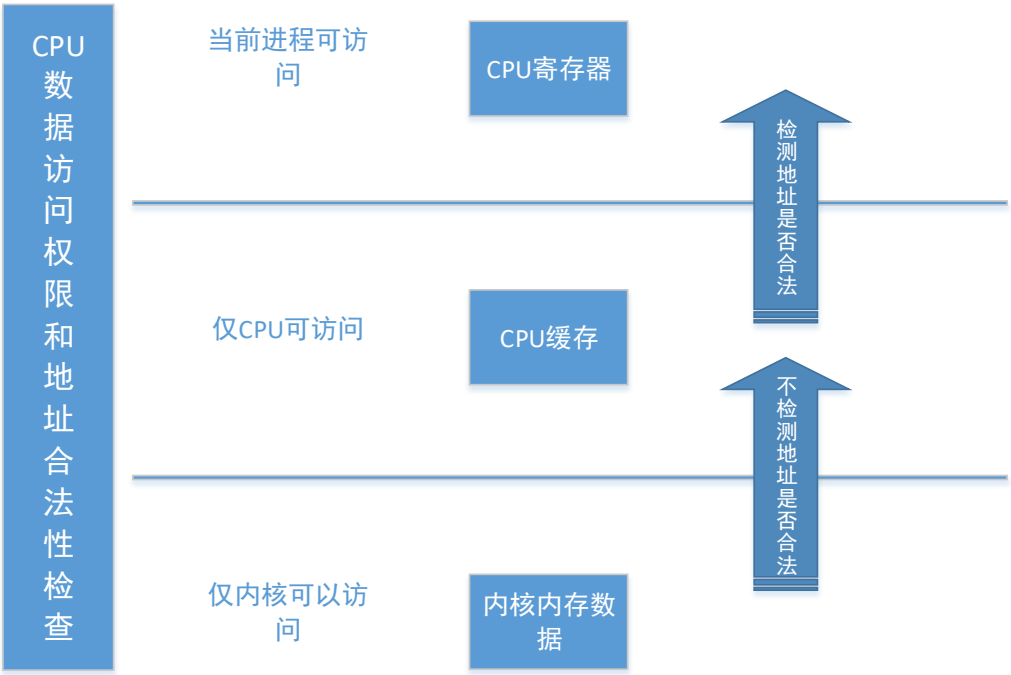

图 3 CPU 数据访问权限和地址合法性检查

如上图所示 CPU 缓存的这个过程对于用户是不可访问的，只有将具体的数据放到 CPU 的寄存器中用户才可操作，同时用户态程序也没有权限访问内核内存中的数据，因此 CPU 采用这种逻辑是没有问题的，但是如果有方法可以让我们得到 CPU 缓存中的数据，那么这种逻辑就存在缺陷。

4.2 边信道攻击缓存

CPU 缓存通常在较小和较快的内部存储中缓存常用数据，从而隐藏慢速内存访问的延迟，缓存侧信道攻击正是利用 CPU 缓存与系统内存的读取的时间差异，从而变相猜测出 CPU 缓存中的数据，结合前边的缓存缺陷部分内容，产生如下的结果：

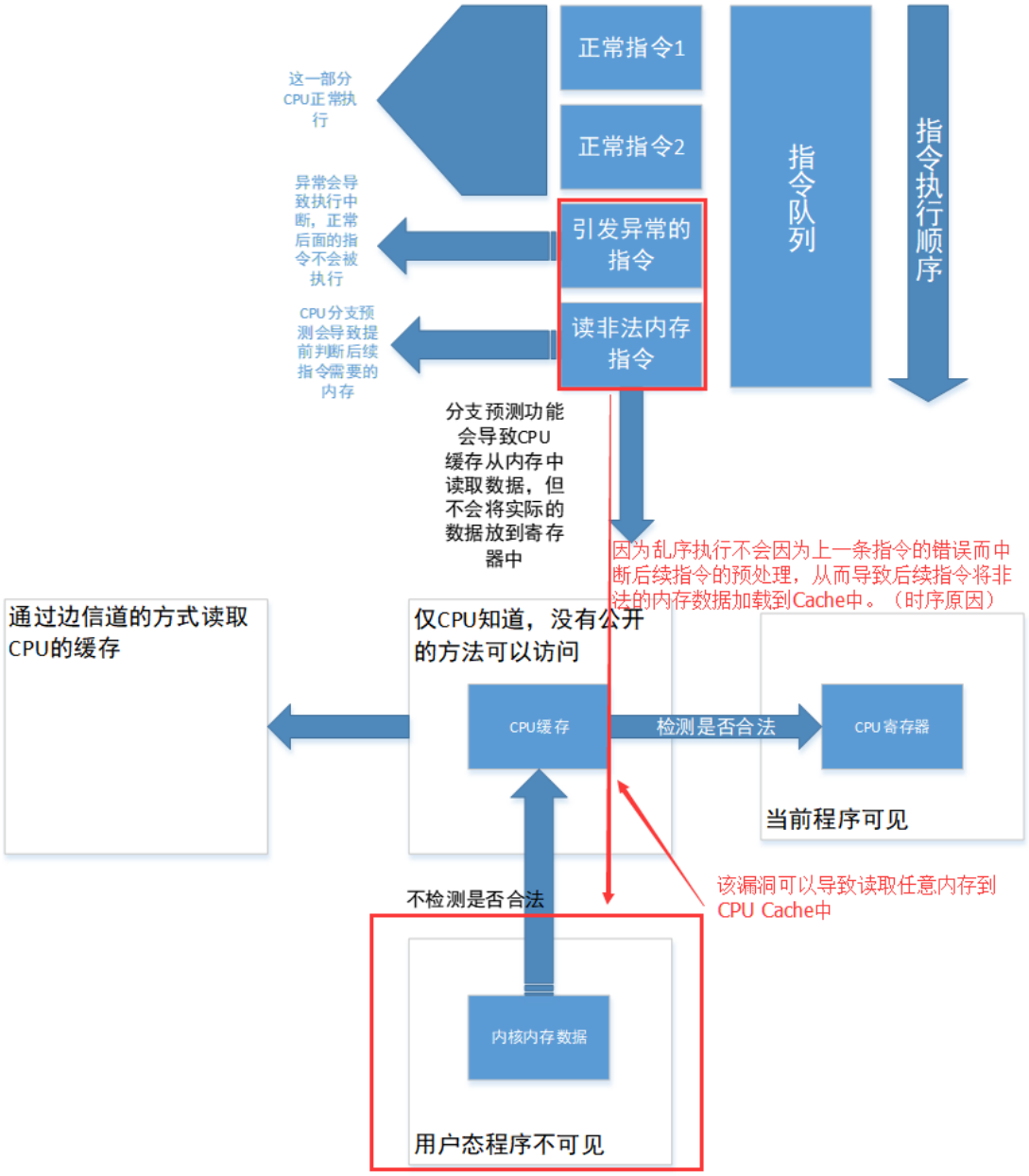

图 4 漏洞原理图

注：简单来说，就是 CPU 缓存中的数据，在用户态和内核态都是无法正常访问的，除非当 CPU 缓存中的数据保存到寄存器中时，会被正常的读取；除此之外，是可以通过边信道的方式读取 CPU 的缓存的。

基于如上介绍的漏洞原理信息，通过 CPU 缓存验证缺陷，并利用边信道攻击技术可猜测 CPU 缓存中的数据，继而访问主机的完整内存数据，造成用户敏感信息泄露。

4.3 漏洞验证

根据《Spectre Attacks: Exploiting Speculative Execution*》报告中包含的 PoC 代码，编译后生成可执行文件，在使用 Intel 处理器的 Windows 7 64 位操作系统实验环境下进行测试。

具体验证细节如下：

- 在内存中申请一段空间，写入如下字符串：

```
char *secret = "Which memory they have visited";
```

- 利用这个机制获取数据

```
Reading at malicious_x = 0000000000000FC0... Success: 0x57='W' score=2
Reading at malicious_x = 0000000000000FC1... Success: 0x68='h' score=2
Reading at malicious_x = 0000000000000FC2... Success: 0x69='i' score=2
Reading at malicious_x = 0000000000000FC3... Success: 0x63='c' score=2
Reading at malicious_x = 0000000000000FC4... Success: 0x68='h' score=2
Reading at malicious_x = 0000000000000FC5... Success: 0x20=' ' score=2
Reading at malicious_x = 0000000000000FC6... Success: 0x6D='m' score=2
Reading at malicious_x = 0000000000000FC7... Success: 0x65='e' score=2
Reading at malicious_x = 0000000000000FC8... Success: 0x6D='m' score=2
Reading at malicious_x = 0000000000000FC9... Success: 0x6F='o' score=2
Reading at malicious_x = 0000000000000FCA... Success: 0x72='r' score=2
Reading at malicious_x = 0000000000000FCB... Success: 0x79='y' score=2
Reading at malicious_x = 0000000000000FCC... Success: 0x20=' ' score=2
Reading at malicious_x = 0000000000000FCD... Success: 0x74='t' score=2
Reading at malicious_x = 0000000000000FCE... Success: 0x68='h' score=2
Reading at malicious_x = 0000000000000FCF... Success: 0x65='e' score=2
Reading at malicious_x = 0000000000000FD0... Success: 0x79='y' score=2
Reading at malicious_x = 0000000000000FD1... Success: 0x20=' ' score=2
Reading at malicious_x = 0000000000000FD2... Success: 0x68='h' score=2
Reading at malicious_x = 0000000000000FD3... Success: 0x61='a' score=2
Reading at malicious_x = 0000000000000FD4... Success: 0x76='v' score=2
Reading at malicious_x = 0000000000000FD5... Success: 0x65='e' score=2
Reading at malicious_x = 0000000000000FD6... Success: 0x20=' ' score=2
Reading at malicious_x = 0000000000000FD7... Success: 0x76='v' score=2
Reading at malicious_x = 0000000000000FD8... Success: 0x69='i' score=2
Reading at malicious_x = 0000000000000FD9... Success: 0x73='s' score=2
Reading at malicious_x = 0000000000000FDA... Success: 0x69='i' score=2
Reading at malicious_x = 0000000000000FDB... Success: 0x74='t' score=2
Reading at malicious_x = 0000000000000FDC... Success: 0x65='e' score=2
Reading at malicious_x = 0000000000000FDD... Success: 0x64='d' score=2
```

图 5 获取内存中的数据

- 获取系统 ntdll.dll 导出函数内存空间数据，下图为 ntdll.dll 的导出函数 NtQuerySystemInformation 的地址空间数据。

```
0x00007FF9060D04F0
007FF9060D04F0  4c 8b d1 b8 36 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 37 00 00 00 f6
007FF9060D0519  04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 38 00 00 00 f6 04 25 08 03 fe 7f 01 75 03
007FF9060D0542  0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 39 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84
007FF9060D056B  00 00 00 00 00 4c 8b d1 b8 3a 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8
007FF9060D0594  3b 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 3c 00 00 00 f6 04 25 08 03
007FF9060D05BD  fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 3d 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd
007FF9060D05E6  2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 3e 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00
007FF9060D060F  00 4c 8b d1 b8 3f 00 00 00 f6 04 25 08 03 fe 7f 01 75 03 0f 05 c3 cd 2e c3 0f 1f 84 00 00 00 00 00 4c 8b d1 b8 40 00 00 00
```

图 6 系统 ntdll.dll 的导出函数 NtQuerySystemInformation 的地址空间数据

- 通过如上方法获取该地址空间数据：

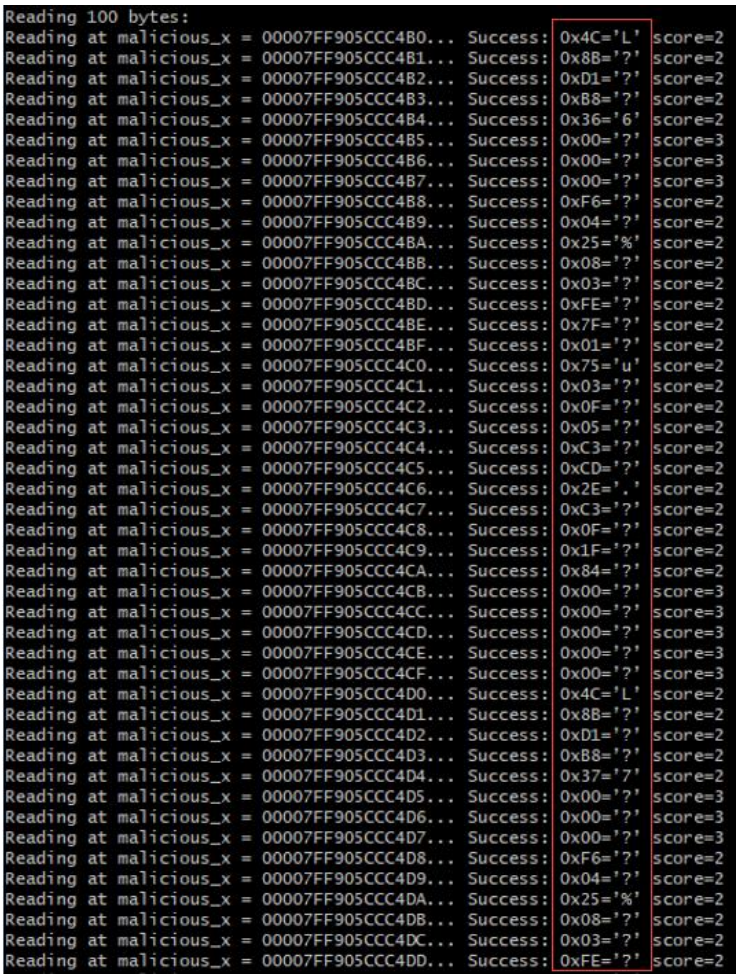

图 7 获取该地址空间数据

通过实验数据证明，该漏洞确实存在，且可被用于获取当前进程内存数据。

5 检测与防御方法

5.1 漏洞检测方法

Windows 用户，通过使用微软公司发布的检测 PowerShell 脚本，能够判断 Windows 系统是否受漏洞影响。

首先，需要安装相应的 PowerShell 模块，对应命令：PS> Install-Module SpeculationControl[5]

其次，需要调用相应脚本，对应命令：PS> Get-SpeculationControlSettings[5]

其中，开启的保护会显示为 True，未开启的保护则会显示为 False，如下图所示：

```
PS C:\Windows\system32> Get-SpeculationControlSettings
Speculation control settings for CVE-2017-5715 [branch target injection]

Hardware support for branch target injection mitigation is present: False
Windows OS support for branch target injection mitigation is present: False
Windows OS support for branch target injection mitigation is enabled: False

Speculation control settings for CVE-2017-5754 [rogue data cache load]

Hardware requires kernel VA shadowing: True
Windows OS support for kernel VA shadow is present: False
Windows OS support for kernel VA shadow is enabled: False


BTIHardwarePresent             : False
BTIWindowsSupportPresent       : False
BTIWindowsSupportEnabled       : False
BTIDisabledBySystemPolicy      : False
BTIDisabledByNoHardwareSupport : False
KVAShadowRequired              : True
KVAShadowWindowsSupportPresent : False
KVAShadowWindowsSupportEnabled : False
KVAShadowPcidEnabled           : False
```

图 8 微软漏洞检测工具运行情况

5.2 载荷应对

安天升级了 AVL SDK 威胁检测引擎，支撑安天自身产品和使用安天引擎用户对相关 POC 的检测能力。

安天产品 IEP 升级了主防机制，对相关 POC 进行拦截。

安天智甲终端防御系统可以对英特尔 Spectre(幽灵)的 POC 进行检测，无论是 Windows 系统还是国产中标麒麟系统。

图 9 安天智甲终端防御系统针对利用处理器芯片漏洞在 Windows 平台系统均可有效防御

图 10 安天智甲终端防御系统针对利用处理器芯片漏洞在 Linux/国产操作系统均可有效防御

6 应对建议

了解相关漏洞的权威文献是阅读论文《Meltdown》和《Spectre Attacks: Exploiting Speculative Execution*》，安天已经在创意安天论坛公益翻译提供了对 Meltdown 的中文译本。

地址：

http://bbs.antiy.cn/forum.php?mod=viewthread&tid=77670&extra=page%3D1

https://spectreattack.com/spectre.pdf

对本漏洞的有效应对核心是大规模的有效的补丁工作，包括提供热补丁。

鉴于该漏洞可以获取完整的内存内容，所以建议相关云设施在完成补丁修复后，应提醒用户修改登陆口令并更新证书。

目前 Linux 内核开发人员已经发布了针对 Meltdown 攻击方式的补丁，详情见：

https://lwn.net/Articles/738975/

Google 部分产品已经针对这些攻击方式做出了修补，但是在部分情况下还需要用户采取额外的步骤确保产品受保护，针对安卓也将在 1 月 5 日推出补丁更新以限制对 ARM 处理器的所有已知变种攻击。详情见：

https://support.google.com/faqs/answer/7622138

微软发布了几个更新来缓解这些漏洞，建议用户在安装操作系统或固件更新之前，请确认防病毒应用程序对其兼容，并且建议用户应用所有可用的 Windows 操作系统更新，包括 2018 年 1 月的 Windows 安全更

新。详情见：

https://support.microsoft.com/en-us/help/4073119/windows-client-guidance-for-it-pros-to-protect-against-speculative-exe

其他厂商也将要或已部分推出针对这些攻击方式的补丁，英特尔也提出正在与包括 AMD、ARM 以及各大操作系统厂商进行合作，以制定行业规范方法，及时、建设性的解决这个安全问题。

目前部分厂商已针对此漏洞做出反馈：

VMware：

https://www.vmware.com/us/security/advisories/VMSA-2018-0002.html

AMD：

https://www.amd.com/en/corporate/speculative-execution

Red Hat：

https://access.redhat.com/security/vulnerabilities/speculativeexecution

Nvidia：

https://forums.geforce.com/default/topic/1033210/nvidias-response-to-speculative-side-channels-cve-2017-5753-cve-2017-5715-and-cve-2017-5754/

Xen：

https://xenbits.xen.org/xsa/advisory-254.html

ARM：

https://developer.arm.com/support/security-update

Amazon：

https://aws.amazon.com/de/security/security-bulletins/AWS-2018-013/

Mozilla

https://blog.mozilla.org/security/2018/01/03/mitigations-landing-new-class-timing-attack/

附录一：参考资料

[1] 《Kernel-memory-leaking Intel processor design flaw forces Linux, Windows redesign》

https://www.theregister.co.uk/2018/01/02/intel_cpu_design_flaw/

[2] 《Google: Almost All CPUs Since 1995 Vulnerable To "Meltdown" And "Spectre" Flaws》

https://www.bleepingcomputer.com/news/security/google-almost-all-cpus-since-1995-vulnerable-to-meltdown-and-spectre-flaws/

[3] 《Meltdown and Spectre》

https://spectreattack.com/

[4] GitHub 上用户 ID 为 turbo 提供的 PoC 代码下载地址

https://github.com/turbo/KPTI-PoC-Collection

[5] 微软提供的检测方法

https://support.microsoft.com/en-us/help/4073119/windows-client-guidance-for-it-pros-to-protect-against-speculative-exe

[6] 分支预测

https://baike.baidu.com/item/%E5%88%86%E6%94%AF%E9%A2%84%E6%B5%8B/9252782?fr=aladdin

[7] 乱序执行

https://baike.baidu.com/item/%E4%B9%B1%E5%BA%8F%E6%89%A7%E8%A1%8C/4944129

附录二：关于安天

安天是专注于威胁检测防御技术的领导厂商。安天以提升用户应对网络空间威胁的核心能力、改善用户对威胁的认知为企业使命，依托自主先进的威胁检测引擎等系列核心技术和专家团队，为用户提供端点防护、流量监测、深度分析、威胁情报和态势感知等相关产品、解决方案与服务。

安天的监控预警能力覆盖全国、产品与服务辐射多个国家。安天将大数据分析、安全可视化等方面的技术与产品体系有效结合，以海量样本自动化分析平台延展分析师团队作业能力、缩短产品响应周期。安天结合多年积累的海量安全威胁知识库，综合应用大数据分析、安全可视化等方面经验，推出了可抵御各类已知和未知威胁的多样化解决方案。

全球超过一百家以上的著名安全厂商、IT 厂商选择安天作为检测能力合作伙伴，安天的威胁检测引擎目前已为全球近十万台网络设备和网络安全设备、超过八亿部移动终端设备提供安全防护，其中安天移动检测引擎是全球首个获得 AV-TEST 年度奖项的中国产品，并在国际权威认证机构 AV-C 的 2015 年度移动安全产品测评中，成为全球唯一两次检出率均为 100%的产品。

安天技术实力得到行业管理机构、客户和伙伴的认可，安天已连续五届蝉联国家级网络安全应急服务支撑单位资质，亦是中国国家信息安全漏洞库六家首批一级支撑单位之一。

安天是中国应急响应体系中重要的企业节点，在红色代码 II、口令蠕虫、震网、破壳、沙虫、方程式、白象、魔窟等重大安全事件中，安天提供了先发预警、深度分析或系统的解决方案。

安天实验室更多信息请访问： *http://www.antiy.com（中文）*

http://www.antiy.net（英文）

安天企业安全公司更多信息请访问： *http://www.antiy.cn*

安天移动安全公司（AVL TEAM）更多信息请访问： *http://www.avlsec.com*