**我对Spectre和Meltdown的验证和理解**

测试环境：

CPU：Intel core i7-3770

系统：Win7 64，Centos7 64

1. 我验证了什么？

<1> Meltdown可以越权读内核内存（缓存中）

<2> Spectre 可以读当前进程内存

<3> Spectre 可以越权读内核内存（缓存中）

<4> Meltdown 可以在已知版本linux上bypass KASLR（几秒，100%）

<5> Spectre可以在已知版本linux上bypass KASLR（几秒，100%）

<6> Spectre + Meltdown混合可以越权读任意内核内存

<7> Spectre可以越权读任意内核内存

<8> Spectre可以跨进程读内存

1. Spectre与Meltdown的异同点？

共同点：

都利用了瞬态指令（Transient instruction）的副作用（缓存）

都是用边信道（timing)获得泄漏信息

不同点：

Spectre利用预测执行（speculative execution），Meltdown利用乱序执行（out-of-order execution）

Meltdown只对Intel CPU有效

Meltdown触发trap

Spectre更难防范

另：

官方文档认为Spectre与Meltdown功能是正交的，Spectre不能越权。其实Spectre也可以越权。Spectre的能力包含了Meltdown。

1. 攻击场景？

Meltdown 可用于越权读内核内存，且存在潜在提权可能。

Spectre由于不会触发trap，可以用于浏览器沙箱攻击，以及虚拟机guest/host攻击，及其它存在代码注入（如JIT）程序的潜在攻击。又可用于越权读内核内存，且存在潜在提权可能。

目前外界尚没有公开信息说明Spectre具有越权能力。

目前外界尚没有公开的PoC支持任意内核内存读取，官方文档未包含读任意内存的技巧。从官方文档中提取到的已知信息只能成功读取缓存中的内存信息。

1. 如何防范？

KPTI可以防范Meltdown，但性能损失严重。

目前我想到的一个几乎不损失性能的方法，已验证能够防范经典Meltdown攻击。是否能防范全部变种还在评估中。已知此方法不能防范Spectre变种。

对于Spectre，除提升flush指令特权等级外，目前想不出完善的方法。

（这篇不是技术分析文，技术分析敬请期待下一篇）

= = = = = = = = = = = =

续

= = = = = = = = = = = =

鉴于网上已经有很多别人的分析了，重复内容将被精简，只保留最重要的核心内容，以及与网上不同的部分。

1. Meltdown

精髓就在于原论文(1)中这个指令序列：

1 ; rcx = kernel address

2 ; rbx = probe array

3 retry:

4 mov al, byte [rcx]

5 shl rax, 0xc

6 jz retry

7 mov rbx, qword [rbx + rax]

3、6句可省略，实测对成功率无影响，

4 句读内核内存失败完成之前，5、7句以瞬态指令模式被执行，

影响到用户内存的缓存状态，之后可以通过边信道攻击取回信息。

论文和网上都没有指出，目标内存需要提前在缓存中，才能实施成功。论文说能读任意内存，任意物理内存，其实只有在缓存中的才能成功读取。

实践发现，通过Spectre与Meltdown混合使用，才真正可以读取任意内存。另外二者混用，还可以避免Meltdown触发页错。

单独Meltdown只能实现bypass KASLR，以及读取部分内核内存。

1. Spectre

V1，

能力是越界访问

精髓就在于原论文(2)中的：

if (x < array1\_size)

y = array2[array1[x] \* 256];

x 越界后，y = array2[array1[x] \* 256]; 依然以瞬态指令模式被执行。

影响到用户内存的缓存状态，之后可以通过边信道攻击取回信息。

论文认为Spectre和Meltdown能力正交，不能越权，其实Spectre具有Meltdown的全部能力。而且Spectre不需要和Meltdown混用，就可以读取任意内存。

V2，

能力是共核跨进程污染间接分支

这是网上唯一还没出现PoC的版本

精髓在我丑陋的代码片段里：

extern char hyper\_buf[];

void shared\_lib(void)

{

in\_lib\_table\_func();

}

攻击进程通过COW把自己进程空间的lib函数表（\*in\_lib\_table\_func）修改成一个搜索到的gadget位置，去训练预测跳转器，使受害者进程执行设计好的路径，把数据（对cache的影响）交出去（e.g. ROP {…; y = hyper\_buf[x];}）。再通过边信道攻击取回信息。

这个攻击理论上最大威力能达到：

1. 本地普通用户监听（进程）盗取root用户密码。
2. 受害者浏览一个网页，微信（跨进程）聊天数据被盗取。
3. 攻击者通过不断创建、删除云主机，遍历物理机，获取海量受害云用户信息。

但这种攻击实现条件极其苛刻，基本只有实验环境才可以实现。

(1) Meltdown https://meltdownattack.com/meltdown.pdf

(2) Spectre https://spectreattack.com/spectre.pdf