EN ESTE ARTICULO >>>

Meltdown
Spectre
Fallo en la arquitectura de procesadores Intel

Dos grandes vulnerabilidades potencialmente peligrosas

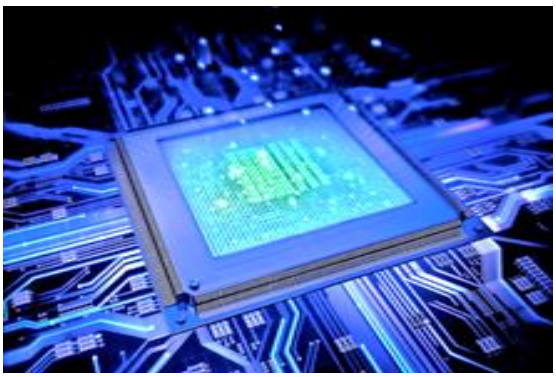
Meltdown & Spectre

Autor: Juarez Aguilar Osmar

Procesadores Intel>>>

Intel Pentium es una gama de microprocesadores de quinta generación con arquitectura x86 producidos por Intel Corporation. El primer Pentium se lanzó al mercado el 22 de marzo de 1993,1 con velocidades iniciales de 60 y 66 MHz, 3.100.000 transistores, cache interno de 8 KiB para datos y 8 KiB para instrucciones; sucediendo al procesador Intel 80486.

El Pentium Pro es la sexta generación de arquitectura x86 de los microprocesadores de Intel, cuya meta era remplazar al Intel Pentium en toda la gama de aplicaciones, pero luego se centró como chip en el mundo de los servidores y equipos de sobremesa de gama alta.



¡Vulnerabilidades por un fallo en la arqitectura!

En el año de 2017 se dio a conocer un par de vulnerabilidades que atacaban usando problemas con el diseño de la arquitectura de los procesadores. Quienes las dieron a conocer fue un grupo de ingenieros que trabajan en un proyecto con Google. Este grupo de investigación es denomindado Project Zero, el cual se encarga de detectar problemas de seguridad en diferentes sistemas para que se tomen medidas y se solucionen y así hacer una seguridad informática libre de errores y vulnerabilidades.

Fue este grupo quien detecto dos errores de seguridad y los denominaron como Meltdown y Spectre. De los cuales hablaremos mas a detalle sobre estos y sobre como funcionan.

Para entender que son y como funcionan todo se remonta desde el diseño de un procesador Intel, hace mas de 20 años.

Así es, pues el procesador Pentium de Intel salió a la venta en 1993 y este procesador incluía El Controlador de Interrupciones Programable Avanzado Local x86 (LAPIC, o conocido solo como APIC). El cual es el que se encarga de administrar los eventos de interrupción enviados al procesador.

En un principio el APIC era un circuito separado, pero este se integró en la microarquitectura del Pentium para permitir un acceso rápido y mas flexibilidad en la administración de APIC, los registros del chip se asignaron a la memoria del procesador en la región de 4KB entre 0xFEE00000 y 0xFEE01000.

Esto causó inadvertidamente conflictos con el software el cual usaba este rango de memoria para otros fines. Para resolver este problema, la familia de procesadores Pentium Pro extendió el APIC para permitir la reasignación de los registros a otra región de la memoria. **[1].** Esta capacidad corrigió un problema raro con el software, sin embargo, consiguió que se genera otro poblema mucho mas importante y peligroso.

Anillos o Niveles de seguridad >>>

Anillo -1 (Hypervisor)
Usualmente en este nivel de seguridad es donde reside el *Hypervisor*, también llamado Virtual Machine Monitor, el cual es el que crea una plataforma virtual en el host de la computadora, encima de donde se ejecutan y monitorean múltiples sistemas operativos. De esta manera múltiples sistemas operativos, los cuales también son múltiples instancias de el mismo sistema operativo o de diferentes sistemas operativos, pueden compartir los recursos de los hardware ofrecidos por el mismo host. [2]

Anillo -2 (SMM)
El anillo -2 es comúnmente llamado System Management Mode. Este nivel de seguridad es el que realmente está a cargo del procesador, es decir este nivel de seguridad es el que se encarga de controlar el hardware, firmware y la mayoría de las validaciones criticas de seguridad.
El modelo de seguridad SMM se basa sobre la premisa de una región de la memoria segura y protegida, la RAM de gestión del sistema (System Management RAM). En concepto, el código del SMM reside exclusivamente en la SMRAM, y solo se puede acceder a esta mientras el procesador está en SMM. Quien controla este acceso a este pedazo de memoria restringido es el Memory Controller Hub (MCH), ubicado entre el núcleo del procesador y la memoria. Si el procesador no está en SMM, el MCH bloquea el acceso a la SMRAM. [3]

La capacidad de reubicar los registros APIC introduce una vulnerabilidad compleja en el Modo de Gestión del Sistema (SMM). Si la ventana de registro del APIC se mueve para superponerse al rango definido para la SMRAM, los accesos de memoria que deben enviarse al MCH para su validación son aceptados prematuramente por el APIC, y nunca recibidos por el MCH. Esto proporciona a un código de anillo 0 una pequeña influencia indirecta sobre el SMM y viola la separación arquitectónica fundamental de los dos modos de ejecución. [1]

Esta vulnerabilidad había estado por mas de 20 anoosos en nuestro procesador y nadie se había dado cuenta. Muchos han hablado sobre el tema y uno de ellos es Christopher Domas, el cual es un investigador de seguridad cibernética e ingeniero de sistemas, que actualmente investiga la explotación de procesadores de bajo nivel. Él es mejor conocido por liberar soluciones poco prácticas a problemas inexistentes. En su articulo sobre *El Agujero Negro de la Memoria* (*The Memory Sinkhole)*, en donde explica detalladamente como funciona esta vulnrabilidad de los procesadores x86. [4]

Pero, ¿cómo se relaciona este fallo en la arquitectura con las vulnerabilidades encontradas por Project Zero? Bien enseguida hablaremos de esto, sin embargo, para entender un poco mejor estas vulnerabilidades es neceasrio abordar dos conceptos importantes en los que se basan estos malwares.

El primer concepto a tener en cuenta es la **ejecución fuera de orden** la cual es una característica de rendimiento muy importante en los procesadores de hoy en día para que estos puedan superar las latencias de las unidades de ejecución ocupadas, por ejemplo, una unidad de recuperación de memoria debe esperar la llegada de los datos de la memoria. En lugar de detener la ejecución, los procesadores modernos ejecutan las operaciones fuera de orden, es decir, miran hacia adelante y programan las operaciones que van a continuacion a las unidades de ejecución inactivas del núcleo. Sin embargo, tales operaciones a menudo tienen efectos secundarios no deseados, por ejemplo, las diferencias de tiempo pueden filtrar información tanto de la ejecución secuencial como la que está fuera de orden. [5]

El otro concepto es la **ejecución especulativa** cuya idea consiste en llevar a cabo un trabajo antes de saber si será realmente necesario con la intención de evitar el retraso que supondría realizarlo *después* de saber que sí es necesario. Si el trabajo en cuestión resulta ser innecesario, la mayoría de los cambios realizados por ese trabajo se revierten y los resultados se “ignoran”. Aunque realmente todo este trabajo innecesario se va directamente a la memoria chache.

El objetivo de esta técnica es proporcionar una mayor concurrencia en caso de disponer de más recursos. Esta técnica se utiliza en una variedad de áreas informáticas, incluyendo la predicción de saltos en las CPU que soportan segmentación, la predicción de valores dirigida a explotar la localización de valores,[6] la prelectura de memoria y archivos, y el control de concurrencia optimista en sistemas de bases de datos. [7]

Durante la ejecución especulativa, el procesador realiza conjeturas sobre el resultado probable de las instrucciones de bifurcación. Las mejores predicciones mejoran el rendimiento al aumentar el número de operaciones ejecutadas de forma especulativa que se pueden realizar con éxito.

¡Ahora si!, hablemos de Meltdown y Spectre

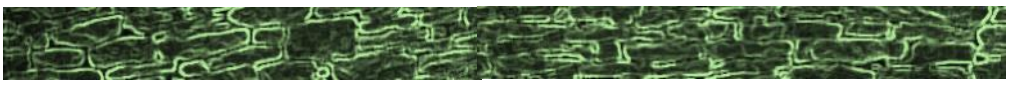
Meltdown es un ataque explota los efectos secundarios de la ejecución fuera de orden en los procesadores modernos para leer ubicaciones arbitrarias de memoria del kernel, incluidos datos personales y contraseñas. La ejecución fuera de orden es una característica de rendimiento indispensable y está presente en una amplia gama de procesadores modernos. El ataque es independiente del sistema operativo y no se basa en ninguna vulnerabilidad de software. Meltdown rompe todas las garantías de seguridad proporcionadas por el aislamiento del espacio de direcciones, así como los entornos paravirtualizados y, por lo tanto, todos los mecanismos de seguridad que se construyen sobre esta base. En los sistemas afectados, Meltdown permite a un adversario leer la memoria de otros procesos o máquinas virtuales en la nube sin permisos ni privilegios, afectando a millones de clientes y prácticamente a todos los usuarios de una computadora personal. [8]

Meltdown se basa en una condición de carrera del procesador que puede surgir entre la ejecución de instrucciones y la comprobación de privilegios, con el fin de leer sin autorización información mapeada en memoria de una forma detectable antes de que pueda tener lugar la comprobación de privilegios que normalmente impediría que se leyese esa información [9]

Si un proceso intentara leer de memoria no autorizada, la instrucción de lectura en principio llegaría al planificador para ser ejecutada por la CPU, como cualquier otra instrucción. Como es habitual, se elegiría una unidad de ejecución y una unidad del controlador de memoria recibiría la orden de leer el contenido de esa memoria y pasarla a la instrucción, de forma que estuviese lista y accesible de forma rápida en la CPU cuando llegase el momento de ejecutar el resto de la instrucción. En algún momento previo a que se permitiese a la instrucción producir cualquier información de salida, la comprobación de privilegios completaría su trabajo en otra parte. En el caso de una lectura no autorizada, la unidad de ejecución recibirá el dato de que la instrucción no ha superado la comprobación de privilegios — descartará toda la información de la instrucción, nunca le pasará nada al proceso, y abandonará la instrucción para proceder a atender a la siguiente.

En los primeros estados de la ejecución de la instrucción, el planificador de la CPU planificó dos eventos: una comprobación de privilegios, y los primeros pasos de la ejecución de la instrucción. Como parte de esta labor, mientras estaba esperando a que la comprobación de privilegios se completase, la unidad de ejecución empezó su trabajo y solicitó la información. En el caso de un proceso maligno, la información procedía de una dirección no autorizada, pero aun así fue obtenida por el controlador de memoria durante la etapa inicial de la ejecución de la instrucción, aunque fuera después descartada y abandonada cuando la comprobación de privilegios se completó y produjo un fallo.[10]

Normalmente esto no tiene efecto alguno y la seguridad está garantizada, puesto que la información leída nunca se pone a disposición de otros procesos hasta completarse la comprobación de privilegios. Sin embargo, aun cuando falla la instrucción, la información ya ha sido solicitada por la unidad de ejecución y obtenida por el controlador de memoria con el fin de estar listo para procesarla, y si bien la unidad de ejecución descarta la información al fallar la comprobación de privilegios, la caché de la CPU de hecho se actualizó como es habitual al leer información de la memoria, por si acaso esa información fuese necesaria en breve plazo una segunda vez.[10]



¿Qué es un rootkit?

Es un término que se aplica a un tipo de malware, diseñado para infectar un PC, el cual permite al hacker instalar diferentes herramientas que le dan acceso remoto al ordenador. Este malware se oculta en la máquina, dentro del sistema operativo y evade los obstáculos de aplicaciones antimalware o algunos productos de seguridad. [11]

Existen dos tipos de rootkit, el *user-mode y kernel-mode*. Los primeros están diseñados para funcionar en el mismo lugar que opera el sistema operativo y las aplicaciones. Ejecutan sus funciones maliciosas hackeando las aplicaciones del equipo o reescribiendo la memoria que usan dichas aplicaciones. Este tipo de rootkit es el más habitual. En cambio, los *kernel* operan desde el núcleo y proporcionan al cracker los privilegios del equipo más importantes [11]

Ataque de canal lateral

Es un tipo de ataque informático basado en información obtenida gracias a la propia implementación física de una computadora, en lugar de basarse en puntos débiles de un determinado algoritmo. Dentro de estos ataques de canal se encuentra los **ataques al cache** los cuales se basan en monitorizar los accesos al caché realizados por un usuario en un sistema físico compartido, como un entorno virtualizado o algún tipo de servicio en la nube.

La caché de la CPU no es legible por parte de un proceso no autorizado, puesto que es algo interno de la CPU. Pero, usando un ataque coordinado a la caché (una forma de ataque de canal lateral), resulta posible para un proceso maligno determinar si la información de una dirección específica se encuentra en la caché de la CPU[10]

Spectre es una vulnerabilidad que permite a los programas alojados en el sistema operativo del usuario acceder a una dirección arbitraria del espacio de memoria de un programa.

En lugar de una única vulnerabilidad de fácil corrección, el documento de Spectre que publico el grupo de Project Zero describe una clase entera de vulnerabilidades potenciales. Todas las mencionadas en el documento se basan en explotar los efectos secundarios de la ejecución especulativa.

Uno de los efectos secundarios de la ejecución especulativa y en la que mas se basa Spectre es la **predicción de saltos.** En ingles es usualmente llamado **Branch Predicto,** detrás de este cocepto existe una idea muy simple: anticipar o predecir cual de los dos caminos seguir´a un salto: el que corresponde al destino del salto, cuando el salto es tomado o el que contin´ua despu´es de la instrucci´on de salto, cuando el salto es no tomado. Si el salto se predice como tomado adem´as se debe obtener la direcci´on de la instrucci´on destino del salto. En cualquiera de los casos la ejecuci´on puede continuar de forma especulativa a lo largo del camino supuesto. [12]

Durante la ejecución especulativa, el procesador hace conjeturas al resultado probable de las instrucciones de bifurcación. Las mejores predicciones mejoran el rendimiento al aumentar el número de operaciones ejecutadas de forma especulativa que se pueden realizar con éxito.[13]

Las instrucciones de derivación indirectas pueden saltar a direcciones de destino arbitrarias calculadas en el tiempo de ejecución.

Para compensar la flexibilidad adicional en comparación con las ramas directas, los saltos indirectos y las llamadas se optimizan utilizando al menos dos mecanismos de predicción diferentes. Intel describe que el procesador predice:
“Llamadas y saltos directos” de forma estática o monotónica,
• “Llamadas y saltos indirectos”, ya sea de manera monotónica o de manera variable, que depende del comportamiento reciente del programa, y para
• “Ramas condicionales” el objetivo de la rama y si se tomará la rama.[13]

Sin duda alguna Spectre usa mucho mas conceptos que involucran la arquitectura de todo los procesadores, por lo que un ataque de este tipo es realmente dicficil, pero sin duda es el mas peligroso porque un ataque de este tipo puede tomar el control total de nuestra computadora. Hasta el momento solo existen formas de mitigar el ataque de tipo Melrtdown, mientras que para Spectre no hay una manera de solucionarlo y probablemente si hubiera una solución esto significaría un cambio total en la arquitectura de los procesadores y esto podría tardar al menos unos diez años mas.

Nuevo fallo descubierto >>>

Recientemente se ha descubierto un nuevo fallo, parecido a meltdown y spectre. Lo han denomindao Spoiler si quieres saber mas te dejo el siguiente link de una nota al respecto.

- ➔ https://www.theregister.co.uk/2019/03/05/spoiler_intel_processor_flaw/
- ➔ https://arxiv.org/pdf/1903.00446.pdf

REFERENCIAS

[1] Domas, Christopher [20 de Julio 2015] *The Memory Sinkhole.* Recuperado de: https://www.blackhat.com/docs/us-15/materials/us-15-Domas-The-Memory-Sinkhole-Unleashing-An-x86-Design-Flaw-Allowing-Universal-Privilege-Escalation-wp.pdf Consultado el: 14 de marzo 2019. Apartado III y IV Pag 1.

[2] *Introduction to virtualitation* Recuperado de: https://docs.oracle.com/cd/E26996_01/E18549/html/VMUSG1011.html Consultado el: 14 de marzo 2019.

[3] Domas, Christopher [20 de Julio 2015] *The Memory Sinkhole.* Recuperado de: https://www.blackhat.com/docs/us-15/materials/us-15-Domas-The-Memory-Sinkhole-Unleashing-An-x86-Design-Flaw-Allowing-Universal-Privilege-Escalation-wp.pdf Consultado el: 14 de marzo 2019. Apartado II Pag 1.

[4] Black Hat (2015) Recuperado de : https://www.blackhat.com/us-15/speakers/Christopher-Domas.html Consultado el 14 de marzo 2019.

[5] Smith, James E. *Implementation of precise Interrups in pipelined processors* Recuperado de: https://www.isi.edu/~youngcho/cse560m/pinter.pdf Consultado el 18 de marzo 2019

[6] Lazy and Speculative Execution Butler Lampson Microsoft Research OPODIS, Bordeaux, France 12 December 2006. Consultado el 16 de marzo 2019

[7] International Business Machines Corporation. Research Division; Prabhakar Raghavan; Hadas Schachnai; Mira Yaniv (1998). *Dynamic schemes for speculative execution of code* (en inglés). IBM. Consultado el 16 de marzo de 2019.

[8] Lipp, Moritz; Schwarz, Michael; Gruss, et al. (2018) *Meltdown: Reading Kernel Memory from User Space. (PDF)* p.1 sec. *Abstract.* Recuperado de: https://meltdownattack.com/meltdown.pdf Consultado el 16 de marzo de 2019.

[9] Lipp, Moritz; Schwarz, Michael; Gruss, et al. (2018) *Meltdown: Reading Kernel Memory from User Space. (PDF)* p. 6 sec. 4.1. Recuperado de: https://meltdownattack.com/meltdown.pdf Consultado el 16 de marzo de 2019.

[10] Lipp, Moritz; Schwarz, Michael; Gruss, et al. (2018) *Meltdown: Reading Kernel Memory from User Space. (PDF)* p. 8 y p. 9 sec.5.1.Recuperado de: https://meltdownattack.com/meltdown.pdf Consultado el 16 de marzo de 2019

[11] Malencovich Serge, (28 de marzo 2013) *Qué es un rootkit?* Recuperado de: **https://www.kaspersky.es/blog/que-es-un-rootkit/594/** Consultado el 12 de marzo 2019

[12] Aguirre, Guillermo [9 de septiembre 2016] *Predictores de Saltos.* Recuperado de: http://www.dirinfo.unsl.edu.ar/arquitectura2/material/predictores.pdf Consultado el: 17 de marzo 2019.

[13] Paul Kocher, Jann Horn, Anders Fogh, et al. (2018) *Spectre Attacks: Exploiting Speculative Execution. (PDF)* p. 3 y 4 sec. II Backgraund Branch Prediction. Recuperado de: https://spectreattack.com/spectre.pdf Consultado el 16 de marzo de 2019.