Broad Overview

The General Law of Cross-Task Information Leakage

"In any setting where short-term performance optimizations have global effect, a sufficiently clever task can infer the recent history of other tasks by observing its own performance."

Meltdown and Spectre are two hardware vulnerabilities that target the Central Processing Unit (CPU) of computing systems with cross-task information leakage. They were identified by four separate teams of security researchers who then disclosed the vulnerabilities to the company's months prior to public disclosure. These vulnerabilities leverage the performance optimization architecture of modern day processors to their advantage.

- Meltdown allows an attacker to retrieve kernel memory, which is privileged space in your Operating System that may contain password hashes and private keys for data decryption.

- Spectre allows an attacker to retrieve information within the process it operates in, and execute code even if it was isolated in a sandbox within a process.

The vulnerabilities require attackers to already have a way into your network, and at most can exfiltrate data from memory. They are unable to cause things such as privilege escalation, holding your data for ransom, or causing business disruption. They data they retrieve however, may be used for that purpose if it contains the information they are seeking.

The initial rush of patches and microcode (firmware for CPU) updates have been somewhat brute and simplistic in their methods and only seek to mitigate potential practical exploit paths for these vulnerabilities. Researchers do not believe there will potential to fully prevent these types of exploits. These patches enable security functions in the CPU, limit speculation, enable cache tagging, and even reorder the very way in which the processor and OS interact with each other.

These patches have also been found to negatively impact the performance of environments by up to 30%. While the actual impact depends on the hardware and applications installed. Older CPUs created prior to 2014 will have the biggest negative impact. I/O intensive applications have also been found to quantify that impact. To help, Microsoft is allowing some manipulation of what features are enabled, to allow businesses to balance the impact on performance, user experience, and business need.

Changes to the OS to mitigate the risks posed by Meltdown and Spectre have also introduced changes to the operating environment. Antivirus systems, and even some attached hardware must be compliant to reduce the potential for system instability. Applications that commonly benefited from these features will now have lower performance.

Time is necessary in testing the quality of these updates. Intel has had to already pull updates at least twice after negative effects they did not expect. VMWare who jointed microcode with OS updates have also had to revert systems. Microsoft had to pull patches that aimed at AMD processors due to a misunderstanding off architecture that led to systems becoming unbootable. Microcode updates are not typically reversible, making it a risk to install.

While the vulnerabilities are serious, it is important to keep in perspective that the exploitation of these vulnerabilities have sole been in lab environments. The industry is working aggressively to even determine paths not discovered by the original researchers, resulting in additional security updates being released every few days.

The ability to exploit any of these vulnerabilities requires having the knowledge of the target systems architecture as each processor family and generation vary in how they operate. The vulnerabilities alone cannot lead to privilege escalation, hijacking, encrypting files, or even redistribution. They are on their own, only ways to siphon data from areas that the processes should not have access to. This means that in-the-wild attacks using any of these vulnerabilities will be limited, but targeted attacks for high value targets and use by state actors are more probable.

Understanding the Vulnerabilities

Meltdown "Rogue Data Cache Load" / "Processor Covert Channel" (CVE-2017-5754)

Meltdown takes advantage of an efficiency process designed in how the system manages memory mapping between the kernel and user space memory. The table that retained that information is known as the Transaction Lookaside Buffer (TLB). This buffer contains the mappings between virtual memory (used by a process) and physical memory (used by the memory manager). This includes where all the kernel memory was, although is limited in access. This means that when an application made a system call into the kernel, or an interrupt is received, the system can quickly access the memory needed to execute it without having context-switching related overhead resultant from locating the data.

To help protect this data further in case privilege escalation was obtained, Address Space Layout Randomization (ASLR) was employed. It made it more difficult to exploit kernel vulnerabilities because it used randomization to further obscure the location of kernel space information, even though it was privilege limited. It didn't make it impossible however, since randomization was limited to 40 bits.

The way meltdown exploits this process begins with a process. It goes on as follows:

1. This process creates a large user space array that will attempt to go and read a byte of arbitrary kernel memory in a way that will cause the read to happen Out of Order.

2. If the attempt works, the user space array created by the process will briefly contain the secret data (by design) but is flushed before it can be read from that user space, but it is flushed into

the CPU cache. This is the exception.

3. The next step is that this process is then going to iterate through the array elements of CPU cache with a creative instruction, and determine which element cached is local to the CPU (A "cache hit").

4. The space of the cache hit is then pulled from CPU cache and stored in user space. This is known as a "covert channel". This all happens because the CPU cache isn't cleared after exception. Think slow building blocks that may eventually lead to enough data to then gain further access with additional exploitation.

The risk of being able to dump the entirety of kernel memory is possible in Linux and MacOS. For Windows, because of the way the Windows Memory Manager is implemented, it is unlikely, that the entirety of physical memory would ever be mapped into a single process. This means that on an unpatched windows system it is possible that most, but not all the kernel memory, may be able to be leaked into unprivileged areas. This provides an attacker with the potential to identify password hashes or find private keys for data. It's also near-certain to provide a method for container escape in a virtualized environment.

This specific vulnerability is considered specific to Intel, and possibly ARM based processors, because of their failure to clear CPU cache when a page fault or exception is triggered, causing the formation of a covert channel and implemented TSX which allows for hardware transactional memory operations that group instructions which can bypass software exception handling routines.

AMD Processors are not subject to the types of attacks that the current meltdown processors protect against as the AMD microarchitecture does not allow memory references, including speculative references, that access high privileged data when running in a lesser privileged mode when that access would result in a page fault. Research against AMD and other processors remain ongoing.

Patches for Meltdown currently perform two tasks.

First, they split the TBL through Kernel Page Table Isolation (KPTI). This stops kernel memory from being mapped in user processes, which incurs a non-negligible performance impact. Some interrupt handlers are still mapped into user space process, due to requirements.

Second, they enforce flushing of CPU cache between processes as a blunt-instrument, which is where the impact tends to be in performance and how meltdown can obtain data.

Some final notes to take away about Meltdown…

1. The ability to exploit this requires knowledge of the architecture in the environment, and that you already have access to the system. It does not cause privilege escalation on its own.

2. Current patches and microcode updates do not prevent a meltdown attack, but simply mitigate practical exploitation to our available knowledge.

3. Data retrieval using this is fairly slow, and the toll on the CPU would likely be identified by users or active monitoring tools.

4. Performance impacts because of these patches may be reduce over the course of the next 6 months to 1 year as additional security functions are enabled in newer processors. Older processes will not gain this benefit and should be retired.

Spectre (CVE-2017-5715 and CVE-2017-5753)

Speculative execution is a technique used by modern high-speed processors that guess what instruction set is likely to be executed after the current instruction, and prematurely executing them in parallel before it knows which instruction you will run. The idea behind this is that rather than wasting several hundred clock cycles before the processor knows which instruction is next, it creates a checkpoint of its state just before it proceeds to speculatively executive the next steps.

When the value of the initial instruction is returned, the processor then determines whether its guess was correct. If the guess was correct, the speculative execution results are committed with a new checkpoint, and a significant performance increase is yielded. If the guess was wrong, it will discard the results of the incorrect guess by reverting to that checkpoint it created earlier.

These guesses made by the algorithms for speculative execution are referred to as branches. The algorithms "train" based on the current instructions and the history of the instructions executed in the past. The results of these may still be stored in CPU cache until otherwise cleared. The recently chosen paths are stored in the branch target buffer (BTB).

Spectre abuses the branch prediction and speculative execution cycles to leak data via a processor covert channel. It does this by creating a misleading input purposed to make the CPU execute instructions it should otherwise not have executed during correct program execution. The effects of these instructions will be eventually reverted which are called transient instructions.

By an attacker using these transient instruction sequences in a particular order, it then allows them to leak information that gets temporarily stored in the CPU cache before the system clears it through exception thereby exfiltrating the data they are going after.

Spectre can also be used to determine the address of a module in memory and bypass ASLR. To exploit this will likely require some knowledge of the remote program being attacked to cause a collision in the Branch Target Buffer (BTB). This means an attacker must study the victim program to understand the targets and construct an attack program to exploit it.

Spectre can also introduce code in a victim process to access data it should not have access to by manipulating the speculative processes. This means it can escape a sandbox and leak data from elsewhere in the process. This is most useful in situation where we have a browser where one tab may contain malicious code, while another tab contains sensitive information that should not be accessible to the attacker. This includes stored passwords and session keys for websites.

Variant 1 "Bounds Check Bypass" (CVE-2017-5753)

This variant is when the CPU mis predicts a branch and speculatively executes code which then leaks sensitive data into a cache channel. This is because if instructions cause an out of bounds instruction, the CPU may speculative read a value that then triggers a cache load from a secondary value that depends on it. This allows an attacker to then profile the secondary value to determine which cache line was loaded to obtain the original value.

Microsoft has made efforts to resolve this vulnerability by making compiler changes, and making those compiler changes now part of Windows Update. They have also hardened Edge and IE11 to prevent exploit from JavaScript. This was most notably known as the "SharedShadowBuffer".

Compiler changes are ideal to help make a better judgement on which code patterns are likely to be dangerous and inserting serialization instructions on that code to prevent speculation. Heuristics that seek to disallow speculative memory accesses whose address depends on previously speculated fetched data may also limit the practical benefit of this vulnerability.

Variant 2 "Branch Target Injection" (CVE-2017-5715)

This variant works by "training" the CPU's indirect branch predictor to mis-predict an indirect branch into an attacker-controlled destination which can then leak data via the cache.

This variant requires intimate knowledge of the inner workings of the CPU branch prediction implementation on the target system. While this does not mitigate the attack, it does make exploitation more difficult.

Microsoft has changed the kernel to call newer CPU instructions intended to eliminate branch speculation in risky situations. Think the beginnings of the heuristic evaluation mentioned in variant 1.

Intel has employed these additional new interfaces to their CPU's:

- IBRS: Indirect Branch Restricted Speculation.
- STIBP: Single Thread Indirect Branch Predictors isolates branch prediction state between two hyperthreads.
- IBPB: Indirect Branch Predictor Barrier instruction prevents leakage of indirect branch predictor state across contexts (for use on context/privilege switches).

Additionally, Intel is recommending retpolines for [BTI], especially on current processors where that may be faster than the microcode patches for IBPB. Retpolines also require a microcode patch on Broadwell and newer CPUs, presumably because on those even ret ends up being predicted in an exploitable way.

Current patches and microcode updates do not prevent a spectre attack, but simply mitigates some of the practical exploitation paths.

Mitigation

These vulnerabilities are in no way preventable given that they are inherently vulnerabilities against the modern x86 and x86-64 architectures. However, a strong mitigation strategy that employs research of the environments potential risks along with proper patching of OS, Applications, and CPU microcode, along with implementation of appropriate security features for the environment will maximize benefit.

The mitigation strategy must also consider the impact on user experience and balance the needs of users and system integrity.

Understanding the OS Patches

Microsoft has released patches for the following operating systems:

Windows Server 2008 (R2), Windows Server 2012 (R2), and Windows Server 2016.

Windows 7 (SP1), Windows 8.1, and Windows 10 (RTM, 1511, 1607, 1703, and 1709 branches).

No other operating systems from Microsoft Corporation will be receiving the updates. They are considered EOL and should no longer be utilized in the enterprise.

VMWare is updating VC 5.5 – 6.5, ESXi 5.5 – 6.5, Workstation 12.X – 14.X, and Fusion 8.X – 10.X with current support agreements. However, VMWare is integrating the microcode updates with these patches, which has resulted it at least one redaction by VMWare of the patches to avoid system issues.

The Performance Impact

Baseline measurements show a variance of 1% - 15% impact on baseline processing based upon the factors of OS version, CPU family, and the security features employed between the OS and processor. Older systems are unable to support all potential security features, and have the greatest baseline impact.

Applications that function within the scope of user space within the OS, will see the last amount of impact. I/O intensive applications are shown to have an impact of up to 25% over baseline, dependent upon the load and physical components required. This means that while applications that stay mostly contained within the OS or their memory space will see the least effect such as office programs, other programs that rely on multiple forms of resources with every instruction such as networked file management systems, will see the highest potential impact.

Antivirus applications are also impacted, due to the way they have traditionally interacted with system memory. The applications must be compliant to receive OS patches as of January 3rd and beyond, and to avoid system instability.

Legacy hardware attached to networks such as printers, scanners, and other devices that have relied on certain functions that are now restricted by these patches whose drivers are not updated may also result in system instability. There has not been significant reports to date as of the writing of this document.

Observation of network and disk utilization impacts have been mixed, due to the new memory mapping affecting the ability to effectively monitor this on physical systems. This observation problem does not affect virtualized instances where the hypervisor is the middleman to resource transactions.

Current Mitigation Strategy

Monitoring of microcode updates will be the new normal moving forward for the current modern processors. All known fixes only mitigate practical exploit risk, it does not prevent them.

- Identify systems whom have risk factors that multiply their risk, consider for replacement or upgrade.
  (Server 2003, 2008 & 2012 Non-R2, systems running x86 architecture)
- Verify AV is current across all environments.
- Force browser updates for Non-IE in all server environments. ALL browser updates in workstations.
- Update all Windows systems capable of receiving the update.
- Discuss the potential impact of microcode updates for each client and determine scope of deployment.
- Install the microcode, no earlier than February if approved by client. Otherwise, standby for more enhanced microcode changes from vendors.
- Activate the appropriate protection measures.
- Evaluate the results and recommend any infrastructure additions required to balance any impact.

Additional Ways to Mitigate Impact

- Train users to report when their computer is acting unusual or receives error messages they have not experienced before.
- Encourage the use of the latest version of browsers that employ memory isolation and avoid sharing memory buffers between tabs such as Edge, Firefox, and Chrome. These are often the first line defense to reduce risk.

- Disable un-necessary plugins in browsers.

- Use effective web filtering systems to block malicious websites that are frequently updated and part of a "cloud collective".

- Accelerate the deployment of newer workstation OS's if you are running a version no longer supported, or unable to employ all the protective features with minimal impact.

- Accelerate the replacement of hardware that is not 64-bit, more than 4 years old, or have CPU that are in the Intel processor family of Haswell or older. These systems will have the greatest impact in performance from changes and have the most substantial risk.

Hardware & OS Lifecycle Impact and Change Requirements

- Windows XP, Windows Vista, Windows 7 (Non-SP1) and Windows 8 are not compliant and will not be updated by Microsoft. These systems should be upgraded or replaced promptly as they represent the greatest risk.

- Windows 10 Systems that are not any of the versions below, should be immediately updated. (RTM, 1511, 1607, 1703, 1709)

- Systems on CPU hardware that was released prior to 2015 should be considered for accelerated replacement once newer hardware is available. If compliance requires the patches, then upgrade to CPU hardware from 2016 or higher is necessary.

- Antivirus applications that are not current should be immediately removed and replaced with ones that are current and compliant before proceeding with patching.

Microsoft Recommendations

- Windows 10 on 2016-era PCs (Skylake, Kabylake, or newer) have single-digit slowdowns, which likely will not be recognized by the users.

- Windows 10, 8 and 7 on 2015-era PCs (Haswell or older) show potential significant slowdowns that were noticed by users.

- Windows Server on ANY silicon, especially in I/O intensive environments has seen significant impacts when you enable mitigation to isolate untrusted code within an instance. Microsoft highly advises that you evaluate the risk of untrusted code for each Windows Server instance, and balance the security vs performance Tradeoff.

Future Considerations for Protection Moving Forward

The industries response to Meltdown and Spectre are still in their infancy, and there is real consideration that we may only be able to limit, but not prevent these types of vulnerabilities. Here are some currently being considered or in the works.

- Enablement of Process-Context Identifier (PCID) functionality. This would invalidate mappings in the Translation Lookaside Buffer (TLB) based on the process ID. This provides isolation without requiring the pages to be flushed. This would theoretically prevent a process from being able to sense the contents of the cache from another process that is in the TLB.

- Clock Cycle Granularity Fuzzing ("Read Time Stamp Counter"). Exploitation of the vulnerabilities take advantage of a high granularity performance counter in the CPU that provides time down to sub-billionths of a second that allows an exploit to discern with extreme granularity whether some data was in local cache, or had to be retrieved. It already has the allowance to be marked as privileged, but it has never been flagged. This is easier to accomplish in virtualization, then it is to employ in the CPU itself.

- Research is ongoing with the KPTI implementation, as to whether the limited kernel memory that remains mapped for interrupt handlers and similar can be used to dump other offsets in kernel memory.

- Isolation of Browsing Tabs via Process, which may consume more overall memory, but reduce the impact potential of something reading memory.

- Applications that are concerned about this risk may employ "serialized instructions" which obstruct the CPU from predicting the process, thereby avoiding leftover instructions in memory.

- Compilers will be introduced into most modern OS update channels to further help mitigate risk, and applications may be updated as a result.

- Newer processors (Post 2014 builds) may be able to recover some of the performance impact from optimized microcode changes are the response continues.

- Intel may establish use of retpolines for branch target injection, which may also reduce the initial CPU performance impact being observed. This does require code to be recompiled and protects only the programs recompiled.

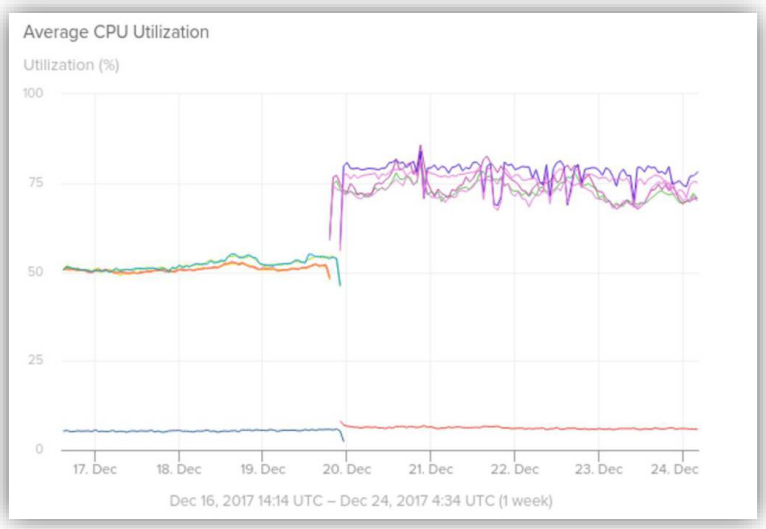
Intel Processor Matrix

CPU/µArch	Spectre – V1	Spectre – V2	Meltdown
i486	N	N	N
Nehalem	Y?[4]	Y[1]	Y?[4]
Westmere	Y?[4]	Y[1]	Y?[4]
Sandy Bridge	Y[3]	Y[1]	**Y**[2]
Ivy Bridge	Y[3]	Y[1]	**Y**[2]
Haswell	Y[3]	Y[1]	**Y**[2]
Broadwell	Y[3]	Y[1]	**Y**[2]
Skylake	Y[3]	Y[1]	**Y**[2]
Kaby Lake	Y[3]	Y[1]	**Y**[2]
Coffee Lake	Y[3]	Y[1]	**Y**[2]
Knights Landing	Y?[4]	Y[1]	Y?[4]
Knights Mill	Y?[4]	Y[1]	Y?[4]
Avoton	Y?[4]	Y[1]	Y?[4]
Rangeley	Y?[4]	Y[1]	Y?[4]
Apollo Lake	Y?[4]	Y[1]	Y?[4]
Denverton	Y?[4]	Y[1]	Y?[4]
SoFIA	Y?[4]	Y[1]	Y?[4]
Lincroft	Y?[4]	Y[1]	Y?[4]
Cloverview	Y?[4]	Y[1]	Y?[4]
Bay Trail	Y?[4]	Y[1]	Y?[4]
Tunnel Creek	Y?[4]	Y[1]	Y?[4]
Stellarton	Y?[4]	Y[1]	Y?[4]

2: Meltdown paper confirms [PRIV-LOAD] on Ivy Bridge, Haswell, Skylake. Sibling microarchitectures presumed vulnerable too.
3: Spectre paper confirms [MISPREDICT],[BTI] on Ivy Bridge, Haswell, Skylake. Sibling microarchitectures presumed vulnerable too.
4: Presumed affected since the issues appear to be pervasive to Intel CPUs and no counterexamples are known yet.

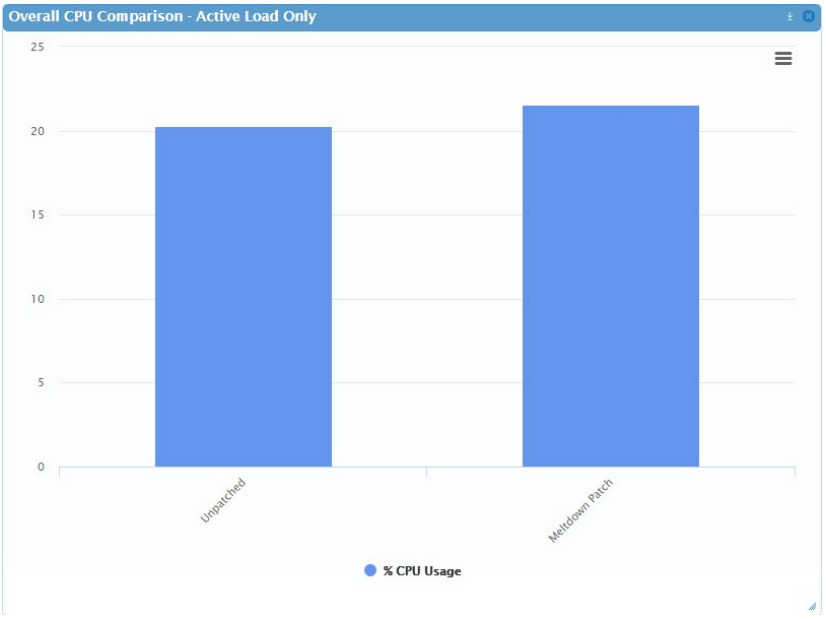
Infographics

SolarWinds Python Code on Paravirtualized AWS Instances (Impact before and after)



Meltdown Only Patch on VMWare ESXi 6.0.0 (No Microcode Change)

Patch Tested	Meltdown ONLY
Hypervisor	VMware ESXi, 6.0.0, 6921384
Guest OS	Windows 10, build 15063
CPU	Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz
Memory	6 GB
Disk	SSD local storage
Density	62 VMs



References and Tools

Important Tools

https://gallery.technet.microsoft.com/scriptcenter/Speculation-Control-e36f0050

https://github.com/ionescu007/SpecuCheck

References and Good Reads

https://arstechnica.com/gadgets/2018/01/heres-how-and-why-the-spectre-and-meltdown-patches-will-hurt-performance/

https://cloudblogs.microsoft.com/microsoftsecure/2018/01/09/understanding-the-performance-impact-of-spectre-and-meltdown-mitigations-on-windows-systems/

https://support.microsoft.com/en-us/help/4073757/protect-your-windows-devices-against-spectre-meltdown

https://support.microsoft.com/en-us/help/4072698/windows-server-guidance-to-protect-against-the-speculative-execution

https://support.microsoft.com/en-us/help/4073119/protect-against-speculative-execution-side-channel-vulnerabilities-in

https://docs.microsoft.com/en-us/azure/virtual-machines/windows/mitigate-se

http://www.felixcloutier.com/x86/INVPCID.html

https://digital-forensics.sans.org/blog/2018/01/08/meltdown-and-spectre-enterprise-action-plan/

http://www.crn.com/news/security/300098051/spectre-meltdown-update-vmware-retracts-faulty-intel-firmware-patches-for-chip-vulnerabilities.htm

https://stratechery.com/2018/meltdown-spectre-and-the-state-of-technology/

https://meltdownattack.com/

https://github.com/marcan/speculation-bugs/blob/master/README.md

https://pdfs.semanticscholar.org/2209/42809262c17b6631c0f6536c91aaf7756857.pdf

https://cloudblogs.microsoft.com/microsoftsecure/2017/06/08/windows-10-creators-update-hardens-security-with-next-gen-defense/?source=mmpc

https://kb.vmware.com/s/article/52345

https://www.vmware.com/security/advisories/VMSA-2018-0004.html

http://www.tomshardware.com/news/intel-bug-performance-loss-windows,36208.html

http://rsec.us/ChromeSiteIsolation

https://arstechnica.com/gadgets/2018/01/spectre-and-meltdown-patches-causing-trouble-as-realistic-attacks-get-closer/